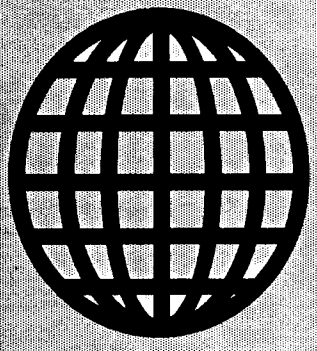
Providing Support Around The World



The Computer Journal

Issue Number 73 May/June 1995 US$4.00

$10 XT

Small System Support

IDE Part 6

8048 Emulator Part III

Small Tools Part II

Real Computing

PC/XT Corner

Dr. S-100

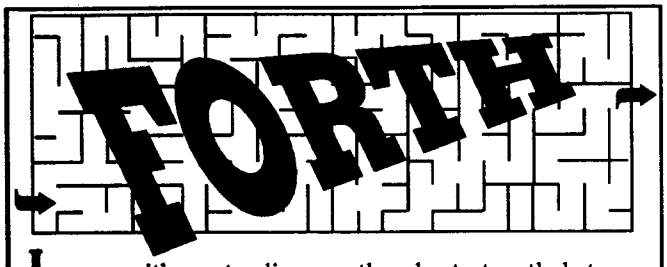
Mr. Kaypro

Centerfold - 640K XT

The Computer Corner

ISSN # 0748-9331

TCJ - For Having Fun With Any Computer!



Journey with us to discover the shortest path between programming problems and efficient solutions.

The Forth programming language is a model of simplicity: In about 16K, it can offer a complete development system in terms of compiler, editor, and assembler, as well as an interpretive mode to enhance debugging, profiling, and tracing.

As an "open" language, Forth lets you build new control-flow structures, and other compiler-oriented extensions that closed languages do not.

Forth Dimensions is the magazine to help you along this journey. It is one of the benefits you receive as a member of the non-profit Forth Interest Group (FIG). Local chapters, the GEnie™ Forth RoundTable, and annual FORML conferences are also supported by FIG. To receive a mail-order catalog of Forth literature and disks, call 510-89-FORTH or write to:
Forth Interest Group, P.O. Box 2154, Oakland, CA 94621.
Membership dues begin at $40 for the U.S.A. and Canada.
Student rates begin at $18 (with valid student I.D.).

GEnie is a trademark of General Electric.

Peripheral Technology Specials

486SLC 33MHZ Motherboard w/ CPU **$119.00**
486SLC/66MHZ IBM,VESA,CPU,Math **$219.00**
IBM board - Made in USA - 3YR warranty

PT68K4/68000/16MHZ /w 1MB	**$249.00**
CDS/68020/25MHZ CPU	**$399.00**
OS9/68000 Includes C Compiler	**$299.00**
420MB Connor IDE Drive	**$215.00**
540MB Connor IDE Drive	**$309.00**
IDE/Floppy/Serial/Parallel	**$24.95**
1.44MB TEAC Floppy	**$49.95**
Panasonic Dual Speed CD ROM	**$159.00**
VGA Card ET4000-1MB,1280x1024	**$99.00**
VGA Monitor WEN .28mm 1024x768	**$229.00**

Free Catalog on Request
UPS Ground $7.00 on most items. Tower & monitor $12.00.

1250 E. Piedmont Rd. 404/973-2156
Marietta, GA 30062 FAX: 404/973-2170

Cross-Assemblers as low as $50.00
Simulators as low as $100.00
Cross-Disassemblers as low as $100.00
Developer Packages
as low as $200.00(a $50.00 Savings)

A New Project
Our line of macro Cross-assemblers are easy to use and full featured, including conditional assembly and unlimited include files.

Get It To Market--FAST
Don't wait until the hardware is finished to debug your software. Our Simulators can test your program logic before the hardware is built.

No Source!
A minor glitch has shown up in the firmware, and you can't find the original source program. Our line of disassemblers can help you re-create the original assembly language source.

Set To Go
Buy our developer package and the next time your boss says "Get to work.", you'll be ready for anything.

Quality Solutions
PseudoCorp has been providing quality solutions for microprocessor problems since 1985.

BROAD RANGE OF SUPPORT

- Currently we support the following microprocessor families (with more in development):

Intel 8048	RCA 1802,05	Intel 8051	Intel 8096
Motorola 6800	Motorola 6801	Motorola 68HC11	Motorola 6805
Hitachi 6301	Motorola 6809	MOS Tech 6502	WDC 65C02
Rockwell 65C02	Intel 8080,85	Zilog Z80	NSC 800
Hitachi HD64180	Motorola 68000,8	Motorola 68010	Intel 80C196

- All products require an IBM PC or compatible.

So What Are You Waiting For? Call us:

PseudoCorp
Professional Development Products Group
921 Country Club Road, Suite 200
Eugene, OR 97401
(503) 683-9173 FAX: (503) 683-9186 BBS: (503) 683-9076

SAGE MICROSYSTEMS EAST

Selling and Supporting the Best in 8-Bit Software

Z3PLUS or NZCOM (now only $20 each)
ZSDOS/ZDDOS date stamping BDOS ($30)
ZCPR34 source code ($15)
BackGrounder-ii ($20)
ZMATE text editor ($20)
BDS C for Z-system (only $30)
DSD: Dynamic Screen Debugger ($50)
4DOS "zsystem" for MSDOS ($65)
ZMAC macro-assembler ($45 with printed manual)

Kaypro DSDD and MSDOS 360K FORMATS ONLY
Order by phone, mail, or modem and use Check, VISA, or MasterCard. Please include $3.00 Shipping and Handling for each order.

Sage Microsystems East
1435 Centre Street
Newton Centre MA 02159-2469
(617) 965-3552 (voice 7PM to 11PM)
(617) 965-7259 BBS

The Computer Journal

Founder
Art Carlson

Editor/Publisher
Bill D. Kibler

Technical Consultant
Chris McEwen

Contributing Editors
Herb Johnson
Charles Stafford
Brad Rodriguez
Ronald W. Anderson
Helmut Jungkunz
Ron Mitchell
Dave Baldwin
Frank Sergeant
JW Weaver
Richard Rodman
Jay Sage
Tilmann Reh

The Computer Journal is published six times a year and mailed from *The Computer Journal*, P. O. Box 535, Lincoln, CA 95648, (916) 645-1670.

Opinions expressed in *The Computer Journal* are those of the respective authors and do not necessarily reflect those of the editorial staff or publisher.

Entire contents copyright © 1995 by *The Computer Journal* and respective authors. All rights reserved. Reproduction in any form prohibited without express written permission of the publisher.

Subscription rates within the US: $24 one year (6 issues), $44 two years (12 issues). Send subscription, renewals, address changes, or advertising inquires to: *The Computer Journal*, P.O. Box 535, Lincoln,CA 95648.

Registered Trademarks

It is easy to get in the habit of using company trademarks as generic terms, but these trademarks are the property of the respective companies. It is important to acknowledge these trademarks as their property to avoid their losing the rights and the term becoming public property. The following frequently used trademarks are acknowledged, and we apologize for any we have overlooked.

Apple II, II+, IIc, IIe, Lisa, Macintosh, ProDos; Apple Computer Company. CP/M, DDT, ASM, STAT, PIP; Digital Research. DateStamper, BackGrounder ii, Dos Disk; Plu*Perfect Systems. Clipper, Nantucket; Nantucket, Inc. dBase, dBASE II, dBASE III, dBASE III Plus, dBASE IV; Ashton-Tate, Inc. MBASIC, MS-DOS, Windows, Word; MicroSoft. WordStar; MicroPro International. IBM-PC, XT, and AT, PC-DOS; IBM Corporation. Z80, Z280; Zilog Corporation. Turbo Pascal, Turbo C, Paradox; Borland International. HD64180; Hitachi America, Ltd. SB180; Micromint, Inc.

Where these and other terms are used in The Computer Journal, they are acknowledged to be the property of the respective companies even if not specifically acknowledged in each occurrence.

TCJ The Computer Journal
Issue Number 73 May/June 1995

Editor's Comments 2

Reader to Reader 3

$10 XT 6
What you can get for $10 at a swap meet.
By Bill Kibler.

IDE Part 6 10
The latest information on the GIDE.
By Tilmann Reh.

PC/XT Corner 13
Frank explains his latest software quandaries.
By Frank Sergeant.

Small System Support 18
C and assembly language tutorial.
By Ronald W. Anderson.

Center Fold 23
PC/XT Schematic.

Real Computing 33
Linux for 386/486.
By Rick Rodman.

Dr. S-100 36
Trenton & letters.
By Herb Johnson.

8048 Emulator 40
Last part of a home built emulator.
By J. G. Owens.

Small Tools 44
Part II about New Micros Inc, F68HC11.
By Calvin McCarthy.

Support Groups for the Classics 46

The Computer Corner 50
By Bill Kibler.

EDITOR'S COMMENTS

Well here we are with issue number 73, the mid year of 1995 already. Time sure flies when your over worked. The last few weeks were very busy, as we had two baby Llamas requiring around the clock care. Lost three weeks of computer hacking, but was able to get #72 in the mail on time. I shouldn't have too many distractions till next year, except for your letters.

We have a short Reader to Reader section because of the mass of material on the PC/XT platform. This is a somewhat special issue that hopefully will enlighten you on some of the good and bad points of using the older PC/XT platforms. Before we start on them a few letters will grab you humor button and interest.

The first of our PC/XT articles is next, as I talk about the $10 XT system I was forced to buy the other day. Yes, forced as you will find out by reading about that event, and how PC's work.

I diverge a little to give you the latest news on the IDE interface that Tilmann Reh is doing for any Z80 platform. We have the facts and a few drawings, also the layout for the serial network card.

Frank Sergeant is back with an update on the PC/XT Corner where he so accurately states many of our readers feelings toward the software deals floating around these days.

We divert again from the PC to make sure you know about C and assembly language programming, as Ronald Anderson gives us his latest installment. Now he has been looking for feedback on these articles and I hope you have dropped him a letter or two. I haven't had a chance to print the glowing letters so many of you have sent on the back of your renewal notices. But we do thank Ron and those who have written to him.

The main feature is next, as this time I am doing the entire schematic of an XT in one issue. Not only is the schematic there, but also the description as well. I have this non-copyrighted manual and thus you get the whole thing. It's 8 pages of information, but for many XT users, it will be there first glimpse of a schematic to their system.

Not all PC's came fully compatible, as we learn from Chuck Stafford in Mr. Kaypro. Chuck covers the 16 bit variants that Kaypro built and hopefully starts you in the right direction to find support.

For those thinking about using their 386/486 machines as real computers, Rick Rodman explains how to load Linux from CD-ROM. I have almost all the CD-ROMs now, got the last set of four for $20 at a swap meet and plan to set up an older 386 with Linux full time (maybe as a Internet server?). Please note that Rick is currently NOT on internet, but trying to find a new provider. I just started using a local service and so try me at tcj@psyber.com (soon to have web page!)

Back to old systems with Herb Johnson, Dr S-100, commenting on the Trenton Computer Show and some letters as well.

I follow Herb with the last installment of J. G. Owens MON48 emulator program. We get the software guide in this issue. Now I have uploaded his files to the DIBs BBS and let me know if you need them elsewhere.

Continuing on with embedded help, Calvin McCarthy had reviewed the New Micros F68HC11 as part of his Small Tools project. So this issues presents what he had to say about their product, and it makes an excellent follow on from last issues "Playing With Micros."

Now I have Brad Rodriguez's part 8, the Camel Forth for 6809, but flat didn't have room. So instead of just two pages, I'll try and get all of it next time.

For those looking for back issues or information on a specific topic, I have asked Rick Rodman to start doing an annotated listing of back issues. I have been trying to do one, but just no time, and as my editorial mentioned last issue went looking for help. Now if you have already done a DB sort of back issues, please ship it to us (or Rick) and we will give you a few free issues for your work.

Since this is a special on PC's, I decided to focus just on troubleshooting help in my Computer Corner. I am currently working on an 8051 development project, but those words of frustration and excitement will have to wait till next time.

Feedback

I thank all the readers who stated how much they enjoy getting *TCJ*, but understand just what pressure I am under. Managing the magazine has ended up being a bit lopsided with almost all my time doing subscription handling. Throw in the occasional Llama raising problem, a wife and kid, job, and wow, what time for doing articles.

I have tried to farm out work, but unfortunately what takes the most time, can not be farmed out. Signing checks and putting money in the bank can only be done by me. Since our numbers have held almost constant for three years, I feel very good about the further of *TCJ*.

A few of the seeds I have been sowing, hopefully will give us many new readers, but should that not happen (or off loading of some work fails), I will certainly be looking for a new editor or publisher. I am giving it a few more tries to make it pay me for my time, but I consider my chances about even at this point. If you can, get your friends to subscribe and just pass the word around about us, we certainly would appreciate it. Thanks. Bill Kibler.

READER to READER

Letters to the Editor
All Readers
MINI Articles

Sub: Feedback on *TCJ*

May 6, 1995
Dear Bill,
After reading your Editor's Comments in *TCJ* #72, I have given quite a bit of thought about your dilemma and about what makes a magazine worth making and reading. For starters, I gathered most of the magazines that come to our house and received quite a shock after I made a list and added the total costs. (Should I stop reading and buy a new car instead?)

It is an interesting subject, and perhaps we can pursue it with your other readers. For now, first let me congratulate you on your 10 years with *TCJ* and let me express my admiration about what you have been able to accomplish with very limited time. Then, for whatever it is worth, I'd like to pass on a bit of my "research" from some of the magazines that I try to read. Actually, I would like to make this a bit of a test.

The following 10 excerpts are examples from 10 different magazines. If anyone can correctly match them with the 10 sources that are listed below, I think a prize would be in order. Let me offer a one-year gift subscription to *TCJ* to the first person who sends me the correct list to my address on GEnie before July 1, 1995. Only one entry per person, and there is a string attached: I would like them to also list the excerpts in order of interest or curiosity, i.e., if they were to choose a magazine on the basis of the sample excerpt, which would be the first one, and so on, in descending order of interest. Here is the test:

*** OUT OF CONTEXT ***

1. Supporting pollution prevention is the focus of life-cycle thinking as an integral part of the well-considered local and global strategy to prevent pollution while maintaining optimal efficiency and effectiveness.

2. As nonbiodegradable waste goes—disposable diapers— containing plastic, paper, and chemical gel, not to mention the special, uh, cargo they were designed to carry—have a half life approaching cobalt's.

3. A couple of the little goats chased after marmots, whose whistling calls echoed between the crags.

4. Intersection now finds the hypothetical intersection of two entities by just picking two entities rather than including an intersection in the aperture.

5. The logical follow-up is a speculative piece sac.

6. It is common to replace GALs with PALs in a production version to reduce cost.

7. Many state-smart definitions can be written using macros to be apathetic about state.

8. In the ideal knee, there is a slight knock-knee arrangement of the quadriceps and the patellar tendon.

9. Durch die Unterbrechung des Laufzeitsystems durch den Garbage-Collection wird das gesamte Zeitverhalten des Systems beeintraechtigt.

10. These words show how the intense study of a single subject, however specialized, can beguile the subtlest of intellects.

A. p.24, *The Computer Journal*, #72
B. p.63, *CADalyst*, May 95
C. p.35, *Forth Dimensions*, May/June 95
D. p.29, *Badminton USA*, Spring 95
E. p.39, *Vierte Dimension*, 3&4/1994
F. p.109, *National Geographic*, April 95
G. p.107, *Automotive Engineering*, February 95
H. p.50, Discover, May 95
I. p.16, *California Chess Journal*, April/May 94
J. p.7, *Invention & Technology*, Spring 95

Bill, you are not the only one with a dilemma. When I can't crawl under the old Volvo anymore, I am going to have to cut out some of these magazines and get me a new car. Now I am waiting for the feedback.
Sincerely, Henry Vinerts

Henry, this is just great, I love it. I will be most interested to see the response to the challenge. Some of your choices are rather interesting. I especially like the German (I believe it is German) sampling.

I think you have hit one area we do try to do our best at, and that is keeping it simple, straight forward, and interesting. Of course we do that by getting people who are actually doing the real work to report about it. Which is why we see so much humor, you have to be a little bit funny in the head (or wherever) to collect, and use these old systems, because it is just plain fun. BDK.

To: B.KIBLER
Sub: Issue #72

Issue #72 was great. The PLD article was really good; particularly striking considering your plaints about keeping the magazine going. Engineers should subscribe; I swear there's more useful information in one *TCJ* than 10 or 20 *EE Times, Embedded Systems Journal,* etc. Of course they're free....

My main client bought me the 80C166 package noted on page 38; we figured it was cheap, since C compilers for this device, and emulation systems, are quite costly. I wound-up writing my own application-specific minimal debugger, after writing a ridiculous program that translates the ASM output of Turbo C 2.0 to 80C166 assembly language.

I particularly enjoyed the Fuzzy Tech demo in the package, although not for any reason related to the 80C166. It was just a good short sweet introduction to Fuzzy concepts, and you can easily pay more than $200 for that. What I learned, incidentally, is that it's like a computer language, *not* a magic new way of thinking of things or an alternative logic, and may be reasonably useful in embedded applications involving control of the physical world. I wrote a letter to *Dr. Dobbs* to that effect, and I suppose it has nothing to do with it, but since then I don't think I've read quite as much drivel about some brave new world where everything will be fuzzy. Now if we can just do the same thing for C—.....

j.g.owen 1-516-421-1807 cis 71121,625 fax 1-516-421-1747, mail02713@pop.net

Well maybe we will see a 80C166 emulator from you as well. Your 8048 finishes in this issue and thanks. It has been fun reading about your tribulations with the machine.

Have gotten a few comments about doing the PLD article, but you must remember I have always said, "PLD's are ok, just give me an alternative." I like Claude's article because he sort of outlined many of the costly options and why it is not for the Saturday hacker to play with. How about some reviews of the 80C166, it is beginning to look like I will never get to play with mine. Bill.

From estabrr@ix.netcom.com
Dear Bill,
First, I'd like to commend you on a job very well done. *TCJ* is my favorite magazine, and it is always the first one I read when the mail arrives. I was quite distressed to read your last EDITOR'S COMMENTS. I'd like to make a few suggestions...

Nothing is more important than your personal health and the well-being of your family. *TCJ* is important to both you and your readership, but not that important. No matter what you finally decide to do, your family and personal life should always come first.

Have you considered changing the publication frequency of *TCJ* from a bi-monthly journal to a quarterly publication? This would reduce your work-load by 1/3, and allow the Contributing Editors a little more time to get their material together. Keep the yearly subscription rate the same. This will raise the price of each issue from four to six dollars, which is still a bargain. You might also consider raising the subscription price to cover the cost of some of the time you spend working on *TCJ*. Time you could be spending with your family, or getting paid for what you do.

Invite one of the Contributing Editors to "take-over" once a year to produce a special issue devoted to some topic of that editor's choice. You would, of course, have to provide some guidance, but this would likely be minimal. Topics might include several articles on a particular kind or type of machine, a favorite operating system, or interviews with computer pioneers. This option, along with the former, would reduce your responsibility from six issues to three.

Expanding the reader base by appealing to a wider audience. *TCJ* might be expanded to include some of the thousands of C-64 and C-128 users out there that now turn to themselves for support now that CBM is dead and its remains have been bought-up by a German firm. These machines have a devoted following, they're easy to repair and modify, and they're inexpensive to acquire. There are also several firms that at one time sold various CBM software and hardware that might still be looking for a good place to advertise.

TCJ might also consider appealing to the growing "Antique" computer collectors movement. I began to acquire various machines several years ago, without any real reasons for doing so. I wasn't until my wife began referring to the shelves full of systems in our garage as my collection that I realized that I too had become a computer collector. Although still in its infancy, this obsession (as my wife refers to it) is apparently catching-on. Appealing to this group could be one way of increasing *TCJ* readership.

As a consumer of computer wisdom rather then a producer of it, I'm not sure what else I can do to solve some of the difficulties you are facing. If, however, you think I could be of some service, please don't hesitate to contact me.

Sincerely yours, Rich Estabrook

Thanks Rich for the ideas, a few might even do. I need to say that I got several notes and letters from other readers saying how they understood my problem and if it becomes family over magazine so be it. One strong point I wanted to make, was I am trying to find someone else to take over in case I must give it up. I have no plans of letting TCJ stop being published, some how and somewhere it will keep going. What needs to be done is just off load a little and get more subscribers. As to publishing fewer, not a chance, almost everyone wants it 12 times a year not less. I already raised my prices to about the maximum, and so get more readers and advertisers is what is needed. That means getting the word

out better and finding more advertisers. The advertisers want a larger subscription base before they will advertise, so first up is more readers.

That bothers me that you think I should cover antiques, since that is what we are supposedly doing. If you think not, then maybe I am falling a little behind in that support. The newest group we want to add is robotic users, since the old units could also be used for that as well.

As to getting help, Rick Rodman offered before, and this time I am taking him up on some help. Also I am going to try and get some Commodore, Apple and Mac writers. So thanks for the comments Rich, and all readers. Bill.

Sub: Address change!
In honor of GEnie finally dropping "geis" from their pathname, I have elected to change my email address. I can now be reached as BJ on GEnie, a.k.a. bj@genie.com on the Internet.

I'm told that any mail sent to b.rodriguez2@genie.geis.com in the last 24 hours may have been lost. If you sent me mail between April 26th and the time you read this message, please send it again. Thanks.

Brad Rodriguez BJ on GEnie
bj@genie.com <===New address!

OK Brad, thought I would run this, as it seems all the services have changed their internet addresses and services. I am also using a local internet provider for now (tcj@psyber.com). Seems like the rush is on for internet. Now all we need is a web browser for CP/M!

I am still looking for someone to provide an internet site with full FTP in and out features. My local service thinks we can do it, but it will cost me both in money and time (the hardest part). Thanks for the articles and support! Bill.

Dear Bill,
This letter is primarily in response to your request for reader feedback at the bottom of your subscription order form.

My primary computers are still 8-bit machines. Of course I use "appliances", like the PC and Mac when I have no choice. But real engineers don' need no steenking 32-bit CPUs and megabyte memories. They can solve any problem with a single IC, programmed right down to the bare metal.

The stuff I design at work uses the Motorola 68HCxx series, Microchip PIC, SGS Thompson, Zilog Z8xx, etc. It's the only way to meet the performance requirements at low cost and high reliability.

How many computers do I have? Dozens: I'm not sure it could be stored in an 8-bit register. Frequency of use? The ones in my thermostat, calculator, alarm clock, kitchen range, and home lighting controller are used all the time. (These are only the products I actually designed, programmed, or built: I'm not counting micros in products that I just use as purchased.

Classic computer usage: This was typed on a Heathkit H89, with my own CPU board. I also regularly use wide variety of other computers. For instance, so far this month I have also used a Kaypro 10, Radio Shack TRS-80 model 100, and SD Sales S-100 computer.

On long vs. short articles: I like longer project articles. But I recognize the need to break them up into several issues for reasons of space, so I don't mind if they are spread out. It's good to also have some "quickie" articles that illustrate simple ideas that others can incorporate into their bag of tricks.

But is's hard to find ones that are of general interest. The tendency is to get rather specialized articles. That makes it less likely to apply to what I'm doing; it becomes primarily educational. So it would be nice to find articles (short or long) that tend to be of more general interest; that tend to pull our diverse interests together.

(Stream-of-consciousness mode on) Speaking of which, at our last Heath/Zenith club meeting, one of the members used INTERSVR.EXE (that comes with DOS v6) to make a PC into a file server, and then ran INTERLNK.EXE on a Z-100. He connected the two computers with a serial cable. The Z-100 (which had only A: and B: floppy disk drives) now had a C: drive: the PC's hard drive as its C: drive.

This is interesting because the Z-100 is NOT a PC clone! While it has an 8088 CPU, it runs a special version of DOS v3. Its memory map, screen, keyboard, and serial and parallel ports are all different than an PC. Yet INTERLNK ran anyway. The only problem was that the sign-on message used the wrong characters for drawing the box border (the Z-100 doesn't have the PC's graphics characters).

This makes me wonder. Could a version of INTERLNK be written for CP/M machines, so they could use a PC as their hard disk drive? This would be wonderful way to enslave a PC to do the bidding of an 8-bit machine!

Meanwhile. I'm still working on an article for you. It tells how to put a surplus Mac motherboard into an Heath H89 computer (or H19 terminal) to build a Mac clone that is actually affordable and expandable.

Yours truly, Lee A. Hart.

Great Lee, can't wait to see that article and thanks for the INTERLNK tip. I just looked (viewed) the interlink files and they are copyrighted 1989/91 by Sewell Development Corp., and they say it will work on any DOS 3.0 machine. They should work on the Z-100s! Might check some old shareware disks and get the original thing?

Hey I got two H19 terminals, so give me the article, I'am ready! Thanks Lee. Bill.

Special Feature

PC/XT Support

What You Get?

$10 XT

By Bill Kibler

For many years now, *TCJ* has been talking about and providing information on PC compatible computers. Recently the price of these units at swap meets has fallen to $10. In this special issue on the PC/XT platform, I felt a review of the PC/XT design and a explaination of what you get for $10 is in order. I'll fill you in on PC technical facts as I recount the how and why I paid the money at the last swap.

The Need

I have been using WordStar since the begining, or at least since I worked for them. In WordStar editing, you use the "control" for moving the cursor, advancing a page, and deleting characters. Since it is second nature for me to just move my little finger over and hit the control key, the newer PC clone keyboards cause me considerable problems. The newer keyboards have the control key any number of places, except where it is most easiest to use. Thus when the last swap meet happened, I ventured forth to fine some replacement keyboards for all my aging and now hard to replace units.

The orginal IBM PC keyboards were extensions of those used on most terminals. The terminals started out using parallel interfaces, and then going to ASCII serial interfaces. Some early keyboards had the terminal CPU doing the looking for key presses. The ASCII keyboards changed to using a small CPU chip, typically 8048, scan for the key press and convert that to the ASCII codes. The codes were then given to the terminal either serially or accross a parallel port.

Somewhere along the line, IBM decided that it would be better if they knew which key was pressed and for how long. This allows for foreign language keyboards, function key operations, multiple key special opertions, and more options than the normal 256 ASCII codes possible. Since serial interfaces are also simpler to work with, the serial interface surfaced as the standard option. So what we have then is two types of PC compatible keyboards. At first there was only the PC/XT variation, typically 83 keys. Later when the AT machines came out to correct all the failings of the XTs a new and larger keyboard (101 keys) came with it (with several modes of scan codes possible).

Scan codes are what the keyboard returns when a key is pressed. There are lots of options, intitalizations, and other keyboard interface considerations, that we will cover at a later date. The reason I was looking for another keyboard, is the newer versions moved the control key location and thus cause me problems. Some earlier keyboards provided alternate keycaps and a switch to allow for selecting caps lock or control key in that location. Many newer boards also have AT and PC/XT option switches as their control codes and handshaking are somewhat different (although I use XT keyboards on AT systems all the time, it really is a ROM BIOS consideration, some support both types, some only one.)

System or Else!

I found a few keyboards, all for $5 each (or less). I found one that was very clean and a real IBM. The price was $10, but I had to take the whole system! That system was professed as a XT with 30 meg hard drive and EGA interface. I really didn't need another XT system, but this one was a true blue unit (all IBM labels everywhere) and thus I thought might be good for my collection. I have also been giving systems to friends to help them get started with computers.

I took the unit home and later started investigating my expensive purchase. The EGA interface was in fact only a COLOR monitor card, the drive was indeed 30 megabytes, but a surprise awaited me. Many times we have discussed what you do when checking out a system. These words of advice hold true whether on a PC/XT or S-100 system. The first step is to open it up and take inventory.

You open it up for more than seeing what is in it. Often in handling and now that many of these units may have been sitting around for years, even under water (like our California winter flooding), you need to see if cards are loose, small dead animals are lying accross traces, and other wonderful surprises. The most common problem would be unsecurred boards that have come loose. I like to pop each board out, see what it is, write that down, check out the socket for dirt and shorted pins. I had one the other day in which the card had been reseated and in doing so, a pin had broken loose and was now wedged against the other side and thus shorting the bus out.

The needed parts

The complete system will have the mother board, at least one video driver card, and a floppy and or hard disk controller card.

Full systems will also have serial cards, parallel ports, network interface cards, and sometimes modem cards.

The early systems (S-100) were composed of many cards and a passive backplane. The cards formed the computer and the backplane allowed for the CPU card to talk to memory and I/O cards. Running memory over these buses has a number of problems (usually speed related) and thus a motherboard design was deamed better. This design limits CPU upgrading, but cuts down on memory interface problems. I/O expansion is handled by adding interface sockets on the motherboard much like Z-100 systems. In the Z-100 you have a motherboard with S-100 expansion sockets (for extra memory, floppy controller, and hard disk controller - 4 slots avaialable).

The major disadvantage of a motherboard design is not being able to upgrade the CPU. In the PC/XT, the expansion slots will take memory cards as well as I/O cards, but at a speed loss. Thus my new system had an expansion card with 256K on it, which added to the 256K on the motherboard to make a 512K system. Now that is about the minimum you can use for most programs, although many of the very early CP/M converted programs will run in 256K.

You will find many PC/XT and most industrial systems designed with passive backplanes and not motherboards. When the designers found ASIC's and PAL's it became possible to build small enough systems that they could fit on a single regular size expansion card (now it can be on a half size card). Several of the Zenith/Heath systems where all expansion cards, which allowed for later CPU upgrades without motherboard replacement.

To Replace or What?

The most common question I get is what to do about "my old but wonderful XT." The typical answer is simply replace the motherboard with a new one. Since the motherboard has become the cheapest part of a system, repairing it seems useless, and if you want to run any of the newer window based software packages, you will also need lots of horsepower and memory by the tons.

You can of course continue to use your old software as many of the older packages work fine and may provide all the computing you need. Often these older units also were just plain flakey and as Dave Baldwin commented in one of his columns, they often came with every other noise reducing capicator missing. I saw a board the other day that had only 5 chips (386SX, 8048, BIOS, EEPROM, ASIC GLUE CHIP) and SIM sockets for memory. The board new would sell for less than $75, and could use all your old XT I/O cards. The re-use of your I/O cards is what most people don't understand.

Just because you might upgrade the motherboard does not mean all the cards must be replaced. You might watchout however that you do not exceed the power supply limits. Many of the first PC/XTs used only 60 Watt supplies and adding a hard disk might overload it. For $40 you can add a new 200 Watt supply and add all the hard drives you want. So for typically $100 you can upgrade you old XT to a new 386 with power supply, using all your old cards and case. Well almost $100 that is, because the new boards take only SIMs and your old unit has DIP memory chips.

RAM Without PINS

Replacing memory chips is often a problem for novice people. They can brake off a pin, bend it under, or put it in backwards and blow the whole chip. The DIP (Dual In-line Package) memory chips use tons of space. Along came Wang and they put the surface mount chips on a small PC board, just a smaller version of the expansion slot idea, and now a full compliment of memory could be on a small half inch by three inch circuit board. Through guides and small drilled holes, alignment and installation is now simple and almost fool proof.

Since almost all the new boards use these sims (mostly now 72 pin not the now obsolete 30 pin) your old dips can no longer be used. Well not exactly, but I doubt many want to shell out the ten to fifteen dollars for dip to sim adapters, as you would need four in some cases, which would also be more chips than you had. If you only need 1 meg of memory, no windows with that amount, you can get 256K sims for about $25 each, or $100 for the 1 meg needed by the 386DX. Now some 386SX units will use a single bank of sims, and thus a single 1 meg sim pack is all that is needed (about $100 or you could use the adapter for $15 and your old chips.)

So the big stumbling block for people wanting to upgrade, is not the motherboard or I/O cards, but cost of RAM. Ever since the fire in Japan that forced memory chip prices up, memory has remained the main cost of upgrading. CPU costs for more powerfull devices has steadily dropped, while memory has remained almost flat, or unchanged. Keeping all the above in mind what other options do we have.

The Surprise.

I said the unit I got for $10 had a surprise, and the surprise is also one way of upgrading. While checking the cards out, I noticed a switch on the back panel. Now normally finding a toggle switch would be for a speed upgrade. We do them often for Kaypros and such, when a new crystal is added along with a higher speed CPU chip. So I looked for a crystal add on and found instead a ribbon cable header in place of the CPU chip. The cable went to a board mounted in the first slot and also the owner of the switch.

Studying it carefully, I noticed a small square chip in the upper corner and on closer examination found a 286 CPU chip. This adapter made the box a "sort of" AT machine, and when the switch was turned off, it didn't work. Further checking found the 8088 socket on the adapter board empty. Normally you would remove the CPU chip, install card and ribbon cable, and put the old CPU chip on the socket on the new board. Thus

allowing the switching between the old CPU and new one. You often needed to do that for some programs that did unmentionable things with the system and crashed at anything other than the older slow speeds.

Well now I was very curious if it worked at all and hunted down my color monitor and some power cables (for $10 you want power cables too!) I fired it up, but no screen and yet if I hit enough keys (F1), it would seem to boot. Hunt for the PC/XT book and check the switches. Oh yes switches. Well you see there are a number of possible options the PC/XT could come with. First off there is memory, 16, 32, 48, and 64K possible and then disks, video, and co-processors?

The switch is based a tried and true method used by S-100 and other systems. Use a small parallel port with a 8 position switch attached. Use the third and fourth one to set the amount of memory. All off is 64K, 3 on is 48K, 3 off & 4 on is 32K, and both on is 16K. Later a second block of switches was added for the 64 to 256K options. 1,4,5 on and 2,3,6,7,8 off is 256K when 64K chips were used.

What did the other switches on switch bank one go for? Switch 2 is for a co-processor or not, on is NOT. 1,7,8 are for disk drives, all on is no drives. The important one for me was switches five and six, which were both off or set for monochrome display. Since I had a color card, I changed 5 to on, 6 off which is for 80 by 25 lines of text on a color adapter card. Powering back up, as these switches are only read during power up, I now saw the error message.

ERRORS

When the unit boots or starts up, it goes through a number of system checks. These are mainly to see how much RAM and what devices are installed, as well as whether or not things are working correctly. In my case I had a parity error in an expansion memory bank. The error indicated bad memory chip and gave me an address in HEX. I changed the hex value to a bank on the expansion card and went "crunching." Ok you say, what is crunching. Well as chips heat and cool, they expand and contract. This action causes them to move in their sockets. The heat also cause some corrison to form on the surfaces of the pins. Between the two actions, you can get poor operation, or often the mysterious parity error. A 9th chip keeps the sum of all bits in memory, and if it not the same when re-checked, something bad has happened(a bit changed on it's own).

The solution is taking the card out and laying it on a stactic free and hard surface and push with you thumb on top of each chip. You should hear a crunching sound as you reseat the chips. Be careful not to over do it and look for old bent under pins which will cause the same problems. I did the same, and each time the location of the error moved, sometimes up and sometimes down in memory. When that happens, I pull the chips almost completely out and then reseat. Often it seems just reseating is not enough to really clean the pins and get good contact again. I did that and zip, 640K of memory was now working properly. I reseated and rechecked all cards and then powered up again and retested all with a big grin on my face. My $10 machines, which in it's own right is a real IBM PC classic system was up and running with a 286 CPU. I tried a few tests and the speed difference was in fact very noticable. Having never tried one of these modificiations, I can now say with experience, that if you can get one cheap enough, do it (it proably is cheaper than what memory is going for these days!)

There are all sorts of upgrades possible, such as V20 in place of the 8088 and that gives you a 8080 or CP/M compatiblity mode. Most of the new upgrade movement these days is turning 386 machines into 486/586 units. Beware however that some very old units had flakey memory circuits, and these newer devices may not like the design. I have a couple of 386 systems, that were tried as 486 upgrades and failed. They still work fine as 386 machines, just not upgradable due to some design variation.

If memory was not the biggest expense, I would just upgrade motherboards on any excuse. Memory however makes some of the 286/386/486 upgrade cards seem almost reasonable. Just don't forget the power supply and that many of your I/O boards will work with any motherboard, even 686s.

About Speed

When people call and ask PC based questions, they are usualy a bit surprised when I stated that many of our newer Z180/280 systems are as fast as 33 MHZ 486's. I suppose now is a good time to clarify that statement. When the orginal IBM PC's came out, they were IBM's third attempt at entering this market. Most engineers who knew anything about the design found it to be very slow and a big step back in computing. I had several memory mapped Z80 systems that would do WordStar about four times faster than early 4 MHZ PC's.

There are two reason for speed differences in this loose comparison. First is how the screen is updated. In PC based machines the memory is BIT mapped. That means a location in memory contains an array of bits, that when moved to the video interface will represent the character presented. It also means that more than one CPU operation maybe needed to move the array from a look up table into the cards screen memory. All these many operations add up to a more complex operation, than a CP/M system using ASCII character mapping. In the CP/M system the ASCII character is taken from the keyboard and stuffed into screen memory, typically one operation. Considerably simpler, but then for plain text why do more.

The reason that higher speed processors still didn't cause very great improvements are based on the ISA (PC standard) bus. The bus was designed with speed limts of 4 MHZ that can be pushed to 8 MHZ tops. So even if you run a 100 MHZ CPU, the fastest you can move data across a ISA bus is 8 MHZ tops. This is also why you have seen so many alternative bus design flooding the market. PC's are crippled machines with far too

much overhead to make efficient use of the components involved. Ask any computer designer and they will tell you that a 68000 based system (like Apple Macs) is far superior with few of the design limits and problems associated with the PCs. So why did people buy them? I have two theories about what went on. The first concept is that large corporations bought units sight unseen. If you were a VP of data processing and the Pres said I want to try these new micosystems, who would you buy from. Most figured that if IBM did it and they failed or worked poorly, who could blame you. In fact I still feel strongly that IBM wanted them to fail, so people would want their old but reliable main frames back.

IBM mis-guessed the market and corporations bought in millions. That made IBM the number one seller and thus John Q public who didn't even know what a computer was for, felt it was safe to buy. That brings me to my second theory, which is their success is based on the fact that most user stole the software from their employer. This wealth of free software then drove the average user to buy a system just like the one at work. Thus more IBMs were sold and they became a stronger number one. Still the product was worse than most CP/M system available at the time, but nobody cared, they just bought and bought more.

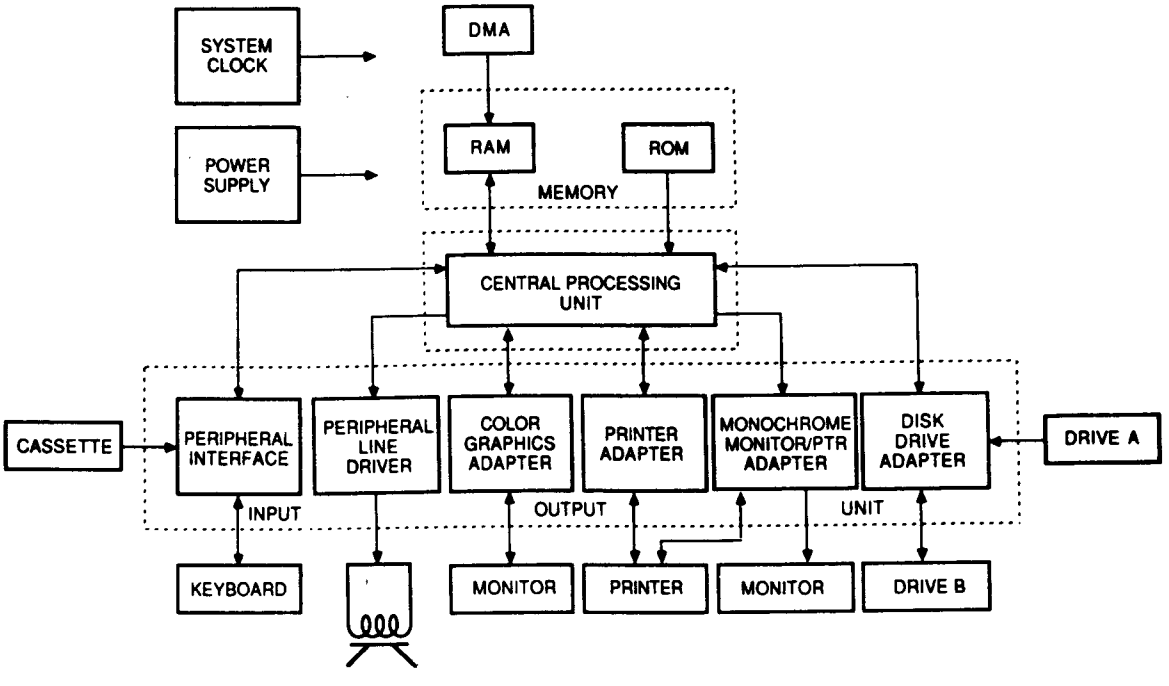
That buying spree carried the USA out of the depression in the eighties, but buyers also became more knowledgable. This learning about how they worked and what you really want in a system, caused the early ninties to see some brakes being applied on sales. So, enter the used car sales mentality, where you need to keep up with the Jones by buying the latest and hotest machine available. Forget that you proably can live without, or your old one actually works fine, new is better and that is all there is to it.

Well not everyone is buying that concept, especially readers of *TCJ*. For many of us, we can see the difference between fast for fast sake and what makes a good design.

All for now

I hope my retelling of my adventure gave you a few pointers and some food for thought. Remember that your system might be working just fine for you now. If so, there is no real need to upgrade to anything else, no matter what the used cars sales people say, whoops, I means computer sellers have to say. If it has died however, determine what died, and if it is the motherboard, replace it with something new. Just because the motherboard gives you some new features, you don't have to use them, unless you have the urge to get frustrated and learn more than you ever wanted to know about computers.

To help see what alternative good designs are avialable, the next issue will have articles about some other machines you could be using. One of those covered will be the PT68K system, running SK-DOS and OS-9. So have fun computing and keep sending those cards, letters, and E-mail. Bill Kibler.



Special Feature

Intermediate Users

IDE Part 6

GIDE

by Tilmann Reh

IDE again, plus words about the LittleNet interface

In *TCJ* issue #71 I already gave a brief description of my current "tiny-hardware" project, the Generic IDE interface board (GIDE). After building up a prototype and getting it running, I was told about some additional wishes of many (yet potential) users. And, as usual with hardware developments, it was necessary to modify the circuit due to a (probably common) oddity of the computer I used for testing the new interface (this was a Triumph-Alphatronic PC-8, a german Z80-based homecomputer running a ROM-BASIC and CP/M-2.2. Hopefully, I will report this development history and the current circuit in more details later - for now, another short description must be enough. As like most developers, I have just too little time for too much work...

Just to give you newcomers an impression about what we're discussing now: The GIDE board is a small (yes, small) PCB which plugs directly into any Z80 CPU socket and contains an I/O-mapped interface for connection to any IDE harddisk drive. The Z80 processor must be removed from the socket and plugged into the interface (daughter-) board. There should be an original-sized drawing of the PCB layout (parts locations) somewhere around this article.

For those that cannot remove the processor because it is directly soldered in, or if there's no sufficient space to plug in the GIDE board, another method of connection is provided. Just solder a DIP socket onto the processor (first case only), and connect it to the interface by a flat ribbon cable which is plugged into a header on the solder side of the IDE interface. However, care should be taken to make that cable as short as possible! And for some computers, it might be necessary to cut a PCB trace on the motherboard and connect a flying lead to the interface board.

Though I can't fully describe the current circuit now, let me say something about my design criterias. In the previous issue of *TCJ*, Claude Palm wrote about PLDs and mentioned that he prefers to put *all* logic circuitry into the PLD. I don't share this viewpoint. In my experience, it is most often better to combine PLDs and "discrete" logic. This has some advantages: First, the result normally is much cheaper (especially for small quantities); second, by using smaller PLDs you get cheaper development tools and are not dependent on any PLD manufacturer; third, you have more PCB layout flexibility since you can locate the small ICs at different locations. I agree that sometimes, especially for large-volume products, it can make sense to use a larger PLD which contains it all - but for this purpose the small ones are definitely better.

The GIDE design uses two standard GALs and two standard 74-series bidirectional registers (74 HCT 646). That's all. For those who want an additional real-time clock (RTC), a socket is provided for it, with a fully decoded chip-select signal. The RTC can be battery-backed by connecting a battery to a 3-pin header. The main design rule was to keep it simple and cheap. Hopefully you agree we met the goal!

One of the two GALs (20V8) does all the address decoding, and provides the necessary chip-select signals for the IDE drive and the RTC. The interface occupies 11 I/O-addresses out of a 16-bitaddress area. The base address (upper four address bits) can be selected by four jumpers. We also could have fixed the base address by GAL programming (it would have been a 16V8 instead of a 20V8 then). But we decided that the user-selectable base address is a real *must* for this interface - we can't provide differently programmed GALs all around the world!

The second GAL (16V8) contains the complete IDE-interface state machine. Its content is roughly comparable to that used in the IDE interface for the ECB bus (described in TCJ #56).

When I first introduced the GIDE project, I included the information that we needed some expressed requests before it makes sense to produce PCBs. There are rather high fixed costs for making PCBs, so a minimum number is required to keep costs reasonable low. From our requests at different networks and user groups, we have about 40 people who are interested in getting a board. Most of them mentioned reasonable prices

they would pay, so here we are: the board will surely be made, and I can offer it at the following prices:

bare PCB	DM 29	$ 22
PCB & programmed GALs	DM 43	$ 32
complete kit, w/o RTC	DM 69	$ 51
complete kit, with RTC	DM 84	$ 62
assembled unit, w/o RTC	DM 84	$ 62
assembled unit, with RTC	DM 99	$ 73

These prices are calculated on the base of DM, assuming today's exchange ratio of about DM 1.35 per US-$ (which is the lowest ratio we ever had!). Probably Herb Johnson will import some bare boards into the USA, and get the other parts there - this will reduce import fees and exchange losses. Hopefully, Herb will be able to offer the same prices and still get some handling fee. And, if the number of orders grows, the price will get lower (remember PCB fixed costs). However, then I need to know this before I order the PCBs (this means: today when you read this!).

If you are interested, or want to place an order, send mail to Herb Johnson, Johnathan Taylor, or me (for addresses see below).

Now lets talk a bit about software aspects. The GIDE interface will physically allow access to the IDE drive, but some software is also needed to put the proper data into the drive's registers and to transfer disk data to and from the drive. I will offer my IDETEST software (which hopefully will be somewhat enhanced soon) which enables direct access to the harddisk drive and also contains some test routines for the drive (like linear and random read/write etc.). This will be good for some fundamental tests of the controller and the drive itself. When all IDETEST checks are passed without errors, the hardware can be assumed to be perfectly running.

You'll then need some software which brings IDE access to your operating system. For CP/M (which is most common for Z80 computers), this means related BIOS routines. I will provide sample routines which show how to initialize and access the IDE drive in terms of physical-sector BIOS routines. It will be up to you to implement them into your BIOS. Especially for the more common computer types, there surely will be someone which does this for the whole community. I (or Herb, or Johnathan) might perhaps serve as a "knowledge-base" of what implementations already are done.

Not all Z80-CP/M computer owners have the sources of their BIOS, or the system generation tools required to create a bootable system from the sources. Then the only reasonable way to support any additional disk device, is by using a loadable driver which relocates straight below the operating system and stays resident there (those PeeCee people call this "TSR" - terminate and stay resident). However, this principle has major drawbacks. The worst is that you can't use anything of the existing BIOS, so the complete deblocking routines and the sector buffers must be implemented again, significantly reducing TPA size. But I hope that someone will write such a generic IDE driver as CP/M-2.2 TSR - I can't provide this by myself since I am using CP/M-Plus only.

Eventually such a generic loadable driver will only be used to check out the new "harddisk feeling" and perhaps will motivate someone to disassemble the original BIOS routines. Once you have the BIOS sources, you are free - everything else is available in the public domain! For keeping track about what's done in this concern, please always inform one of us about current developments!

Finally, some new words about the LittleNet adaptor board:

I promised to do a PCB layout which contains the LittleNet adaptor (an isolated RS-232 to RS-485 converter) which was introduced as a draft in *TCJ* #69. After some discussions with Rick Rodman about the optimum connector types for both sides of the adaptor, we decided to use standard ITT-Cannon DE-9 connectors (one male, one female). The pinout of the RS-232 side will match a standard PC/AT serial connector, and the pinout of the RS-485 side allows for using twisted-pair ribbon cable with crimped connectors (though it's better to use round shielded twisted-pair cable and solder it to the connectors). All adaptors are supplied via the network cable, by YAWT (yet another wall transformer) delivering something between 8 and 20 volts DC.

The original-sized artwork and the placeplan of the single-sided PCB should be printed somewhere near this, along with the current schematics. If there is sufficient interest in PCBs, I could also make a run. If you are interested, contact Rick or me.

Contacts:

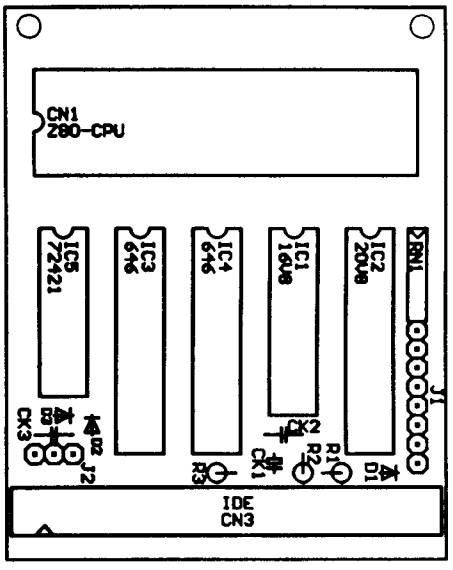
Tilmann Reh, Am Rueckelchen 5a, 57078 Siegen, Germany
InterNet: tilmann.reh@hrz.uni-siegen.d400.de
Fax (at work): +49 271 484520

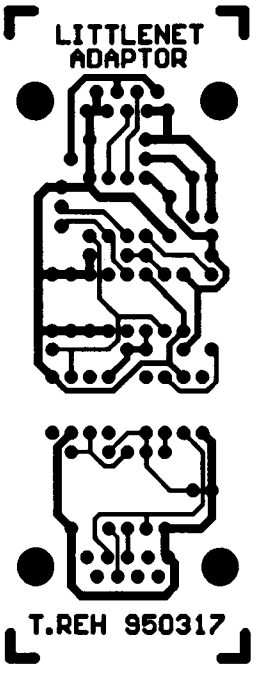
Herbert R. Johnson, CN5256 #105, Princeton, NJ 08543, USA
InterNet: hjohnson@pluto.njcc.com
Voice/FAX +1 609 771 1503 (8am-11pm EDT)

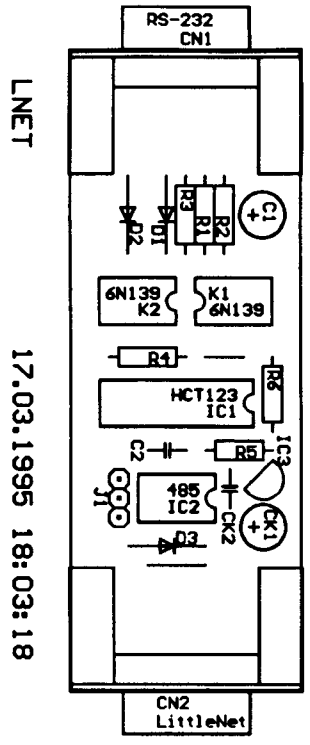
Johnathan Taylor, UK
InterNet: jet@centron.com
Fidonet: 2:2501/307.9

Rick Rodman, USA
InterNet: rickr@aib.com
BBS: +1 703 330 9049 (24h)

GIDE LAYOUT (actual size)

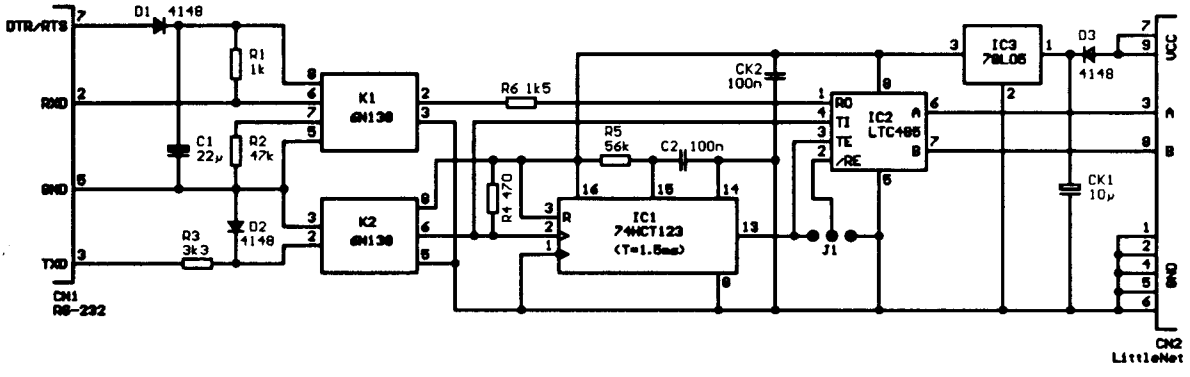






LittleNet isolated RS-485 converter

950421 Tilmann Reh



TCJ Needs Articles

The Computer Journal is looking for new authors to add to our regular writers. Several areas of antique and collectible computers are not currently supported by our regular staff. We would like to add your name to our mast head in supporting some of these wonderful older 8 and 16 bit systems.

Wanted: Articles on Commodore C64/C128, Radio Shack CoCo's, Apple IIe, Macintosh, PC/XT, Timex/Sinclair, Atari ST, Amiga, and any system used for Robotics or any homebuilt computer project.

If you are the sort of person who often explains how your computer works to friends, then you would be perfect for our readers. At *TCJ* we want articles written as if you are just explaining this problem or solution to your friends. It is a one on one conversation, with hundreds of others listening in. There are plenty of readers waiting to hear what you have to say.

Drop those cards or letters to Bill Kibler, *TCJ*, PO Box 535, Lincoln, CA 95648.
(800) 424-8825 or (916) 645-1670
E-Mail: tcj@psyber.com, or B.Kibler@GEnie.com, or B.Kibler at GENIE, or 71563,2243 at Compuserve.

PC/XT Corner

By Frank Sergeant

Special Feature

Hardware Support

I Hate Change

I'm pretty unhappy with the modern PC/Software world. Let me tell you why.

VC++

Because of my favored status (I'm still alive) and my ownership of a competing product, Microsoft has offered to sell me their latest version of VC++ (Visual C/C++) for only $374. It is actually a "subscription." They'll send me the current one now and 3 more over the next year. Am I the odd man out here, or does this sound like the product is unfinished!

Supposedly, I should want VC++ so I can develop for Windows 95 (or, as I've seen on the net, "Windows Winever"). Naturally, I am not expected to object to the system requirements of the product. Now get this: the ad says VC++ requires 16 megabytes of RAM, but 20 megabytes are recommended! (Not only that, but Windows NT is also required.)

Tulips

I think this is like the Tulip Mania, when otherwise reasonable people thought it was perfectly ok for tulip bulbs to increase in price without limit. Only now it is ok for software to be sold that doesn't work right and whose RAM requirements increase without limit. Then, for the privilege of fixing some old bugs and introducing some new bugs, Microsoft (and others) let you "upgrade" for several hundred dollars.

Economics

The problem with change is that things do change, whether I like it or not. In particular, relative values and prices change. Economists think about and argue over whether values are subjective or objective. (Actually, I just made that up. I suppose economists, like the rest of us, only think about how to keep their wives and girlfriends properly separated.) My point is that what we grew up conserving carefully because it was so scarce and expensive, such as RAM, might now be plentiful and cheap. To continue to object to its "waste" as I seem to be doing might be foolish in the mid 1990s. If I could use VC++ to crank out an application that makes me money, what do I care if I have to buy more memory (20 meg at around $40/Mbyte makes the cost of the VC++ compiler look cheap), customers have to buy more memory, the application runs slowly, and it is so complex there is no way to establish that the application works correctly. If there are bugs, I can blame Microsoft.

Watcom

Watcom C/C++ version 10.0 is the competing product I own. I never thought I'd recommend Watcom, and I don't. However, it doesn't look too bad in comparison to the ad for VC++. It runs in only 8 meg (see how the modern software world has corrupted me) and, while their special deal lasts, can be had for about $200. For a Forth programmer, I sure spend a lot of my time writing in C and assembly. Watcom C/C++ has aggravated me a great deal. Its documentation is poor. Its tech support is nonexistent (but, not particularly worse, so I gather, than the tech support from Borland and Microsoft).

Assembly Language

I gave up on the assembler that comes with Watcom (WASM) and went back to Borland's assembler (TASM). Given my druthers, if I had to work in assembly, I would use A86, a great shareware assembler. However, A86 doesn't work in 32-bit mode. I understand that A386 is about to debut, though. Nevertheless, for the big project I've been working on, with lots of large, undocumented modules written in TASM, I felt I had to continue to use TASM. The best way I know to work in assembly language is to do it from Forth. In effect you use Forth as an interactive executing and testing environment for the assembly routines, and write short routines. I have violated my rule on this, to my regret, on this project. Anyway, we eventually grow used to the devil we associate with regularly, so I am coming to tolerate Watcom. You'll be sorry, though, if you use it on a machine slower than a '486 40MHz with 8 meg of RAM.

Tech Support

Let me follow up my comment on tech support. Like Wil Rogers, all I know is what I read on the internet. I have heard nothing but complaints about tech support from Borland and Microsoft. I have experienced the poor tech support from Watcom. Your experiences may be different. If so, I would love to hear about them, and I'll try to update these comments in a future article. However, the one company whose tech support I've heard praised is IBM. The idea was that the others more or less pretend bugs don't exist, but IBM takes them

seriously, puts the reported bugs on a list to be corrected, and keeps you informed of the status of the problem. IBM has been a symbol of "big" that some people have grown to hate (especially Macintosh supporters?). However, I have a fond spot in my heart for them. Part comes from the sensual joy of touching an IBM Selectric keyboard. You press the key, suddenly it gives! Ahhh, so satisfying. Part comes from the appearance, comfort, and reliability of even the first IBM PCs with their beautiful green, long-persistence phosphor monitors.

Vaporware

Feeling this way about IBM and Microsoft, I should buy IBM's OS/2 Warp. I suppose this competes with Windows NT (Microsoft's extreme resource hog professional version of Windows). I think it is "in the air" that OS/2 competes with Windows Winever. However, I'd like to point out that, as I write this, Windows Winever doesn't exist. (Yes it does! It's in Beta! Lots of people have it!) Yeah? If I can't buy it through the big mail order software distributors, Windows Winever is NOT a real product. Which leads me to another complaint about Microsoft. Maybe Windows Winever really will be the greatest thing since sliced fingers, when and if it arrives. However, aren't we seeing a rather underhanded marketing technique? Announcing vaporware far in advance of when they really expect it to be available? I think of this as "reverse fraud" — instead of fraudulently inducing someone to buy your product by lying about your product, you fraudulently induce someone not to buy a competitor's product by lying about your product. I want to emphasize here, in an attempt to prevent any lawsuits against me by Microsoft, that these are strictly my own opinions and feelings. I am not in any way suggesting there is any objective evidence for any of this. Maybe one day I will thank Microsoft for dragging me into the modern world and teaching me effective marketing and RAM usage.

XTs

Some software, however, does run well even on old XTs. I'm writing some of this article with the famous shareware PCWRITE word processor. It is zippy even on an XT. I don't completely like it, but I haven't figured out why. Anyway, it is usable and far friendlier than the "GUI" word processors I've tried, such as Microsoft Word and Novell's Word Perfect, whose modern versions are not usable unless you have a very fast machine. However, Word Perfect's version 5.1 (and earlier) for DOS is very usable. Maybe I'll switch to it. It doesn't take too long to get used to its function keys, even though they have no mnemonic value whatsoever.

Editors

For plain text editing as opposed to word processing, I use the shareware QEDIT. Overall, I'm very happy with it. It is fast and allows multiple files to be open at the same time. I use it for all my batch file editing and all my C and assembly language and Clipper programming work. I hear I would like something like microEmacs. I occasionally try one for a few minutes now and then, but haven't been able to get into it yet. I already know how to do what I want in QEDIT, so I feel a little lost in the others. I was forced to use the vi editor on unix or unix-like systems. I finally got used to it and still use it whenever I work on a unix system. SWT's (Southwest Texas State University's) computer department has adopted emacs as its standard editor. The idea is a good one. Here is an editor available on every PC on campus, on every workstation (i.e. big unix-running boxes), and on the VAX mainframe. Finally, we can standardize and not have to switch editors when we sit at different machines. Instructors only have to teach one editor. Students only have to learn one editor. Did I mention I hate change? Well, I switch between QEDIT on my PC at home to running the EVE editor on the SWT's VAX to running vi on the unix machines. I'm tired of it. Maybe one day I'll either learn emacs or quit using the VAX altogether.

Geos

For a course I took last semester in parallel processing, I needed to draw occasional diagrams as part of my homework. I used various Windows word processors for this. It was unpleasant, but possible. Billy Tanksley (tanksley@san_marcos.csusm.edu) recommended Geos, so I downloaded the sample version. A few days earlier, I had ordered a copy Claris Works for Windows (a combination of word processor, spreadsheet, etc.). I figured if their ad was correct, it might be worth it to have a comfortable word processor under Windows. Well, to make a long story short, I sent the Claris Works back for a refund (which they promptly sent me). Again, I'm not sure why, but I wasn't comfortable with Claris Works, but I was fairly impressed with Geos. There were various limitations in the demo version, but it was at least as easy to crank out my homework text and diagrams as with the higher-priced, larger, more bloated products. Geos doesn't require Windows, it is a replacement for Windows. I hear it runs on all PCs, even old XTs, which certainly is not true about Windows. I haven't tried Geos on a slow machine, so I don't know how much it slows down. Anyway, you can get the demo via ftp from 130.219.44.141 or ftp.mcs.com in / mcsnet.users/jbarr/GeoPublish and see for yourself. I gather the full system is called Ensemble 2.01 and costs around $80 including word processor, drawing program, spreadsheet, and database. Call Geoworks at 800-824-4558 to order. Still, it is "GUI" and I object to "GUI"s.

GUI

While GUI stands for Graphical User Interface, it has come to mean these godawful interfaces such as Windows, where text characters are drawn as if they were complex pictures. I don't like this because it makes the text harder to read and it goes too slow. Also, at least under Windows, the response to what you do (pressing a key) is often delayed. Instant feedback is one of the most important characteristics to me for a usable system. I hate waiting a second or two to see the last few characters I typed because the GUI was busy with something else. I also

hate the usual proportional fonts you usually have to put up with under Windows.

Pentia

Don't buy 'em! (Again, Intel, this is just my own personal opinion!). My experience is limited, but the Pentium processors run too damn hot, and are inherently unreliable because of this. Wait until competitors (or even Intel, perhaps), get the feature size small enough that they run cool. A lot of time has recently been stolen from and worry add to my life because of customer problems that were finally traced to a 90 MHz Pentium computer whose CPU fan didn't always work.

As a joke in the computer architecture labs I used to teach I would say "put your finger on the chip and if it burns a blister, something is wrong! Turn off the power and think about it." Well, call me old fashioned, but if you can't hold your finger on the CPU chip, it's too damn hot, and something is wrong. For the same reason, I worry about the fast '486 chips. My unfounded impression is that AMD's chips might run the coolest.

Complexity

I've said it before and I'll say it again. From a book I read years ago came the idea that sometimes YOU need things, but sometimes you only need things because your THINGS need things. The benefits of simplicity are recursive. If you can use a smaller boat, not only is the boat cheaper, but the anchor you have to buy is also smaller and cheaper.

When it comes to software, I am a Forth enthusiast. Even when you can't use Forth, you can use its ideas. In other words, you can write and test in C or in assembly in a Forth-like manner. Some might just call this good software engineering. Read the book *Thinking Forth* by Leo Brodie (available from FIG (Forth Interest Group, 510-893-6784, fax 510-535-1294, they probably have an ad in this issue, or Miller Microcomputer Services at 508-653-6136 or dmiller@im.lcs.mit.edu) for more of the philosophical details. I don't say I achieve it, but here is my ideal: Forth is small. It doesn't take up much room on your hard disk or in RAM. It comes with full source code. It is interactive. My lovely Pygmy Forth (me? biased?) is about a 16K byte program and runs comfortably in the RAM available on any PC in existence (well, let's say with any PC with at least one eighth of a megabyte of RAM). The executable and full source code fit on a single floppy (with shadow blocks, it may need to be a 720K floppy). Neither Microsoft nor Watcom nor Borland C compilers come with source code. Can you imagine that? And then they charge you for a bug fix they call an "upgrade"? When I come out with a new version of Pygmy, I send the new version to all my registered users at no charge. On the other hand, I don't make as much money as Microsoft.

Anyway, why is smallness important? Why is source code important? Because, if something is small enough, you have a chance to comprehend it completely. Thus, you have a better chance of having your applications work correctly. By including source code, you get full access to the entire system. In Pygmy, you can type VIEW <some word> and pop into the source code for <some word> to see exactly what is going on. You can verify assumptions and correct misassumptions more quickly. If there is a bug in the basic system, you can fix it yourself. You are not at the mercy of a "technical support" department. With full source, you can customize your system as you see fit. You and I do not need to agree completely on what makes a good Forth. The interactive nature of Forth lets you do your experimenting very quickly to zero in on the development problems you face.

Survival

Alright, suppose I agree the above approach is better in some sort of idealistic sense. How does that solve my problems of surviving in the MSW (modern software world)? This is my constant concern. My latest thinking is that I might reconcile Forth and the MSW. My tentative plan is to put my Forth inside a C wrapper whenever I must deal with the MSW. If I don't have to deal with that world, I'll just use Forth directly. Otherwise, I'll let some C vendor, such as Watcom, run interference for me.
One approach is to have a very small C program that allocates an array into which it loads the Forth image. Then, it calls your Forth as a subroutine. When your Forth finishes, it returns to C. C then returns to the operating system. Are there nasty operating system/hardware environment/software environment details you don't want to deal with that the C library can handle? If so, Forth calls those C routines. I'm doing this sort of thing right now under Watcom C with its DOS4GW DOS extender, using my 32-bit Forth (very much like Pygmy, except integers are 32 bits wide, megabytes of RAM are directly addressable, and it has multi-tasking).

There are various other efforts along this line. I find Rob Chapman's **Timbre** and Norm Smith's **Until** particularly interesting. Norm Smith's book *Write Your Own Programming Language Using C++* should be available from FIG. Both **Timbre**and **Until** are available via ftp at taygeta.oc.nps.navy.mil. I think Rob's system lets him write in Forth and have his source code translated automatically to C so it can be compiled under a C compiler. If I followed what he was talking about, this offers a way for a programmer who wants to write in Forth and a manager or client who requires C code to coexist peacefully. I think both Rob's and Norm's systems let you work interactively.

Databases

As if my C/assembly troubles weren't enough, I'm also working in XBase. XBase is the generic term for dBase-like languages. Ashton-Tate's famous dBase is now owned by Borland, but there are various clones and variants. DBase is an interpreter and slow. Clipper was a DBase compiler, but has been extended in various ways, mostly in the direction of the C language. FoxPro is now owned by Microsoft, and is similar

to dBase. FORCE is similar to Clipper. Flagship is a Clipper-clone that runs on Unix systems. Further, there are various C libraries designed to add XBase functionality to C.

A key feature of XBase languages is how the .DBF file works. A database file (e.g. CUSTOMER.DBF) include the details of the file's record and field structure inside the file itself. This adds an interesting independence to the data. Such database files can be read and manipulated by various utilities or other XBase languages, without necessarily requiring the original application program that created the files. The record layout is included within the database file. This is great if you are a customer and are changing vendors. The different XBase languages typically have their own proprietary index file format, so you just move the database files and let the new system re-index them however it prefers. There is an entire XBase world and culture that felt pretty strange to me as I first entered it. Actually, I met George Tate (founder of Ashton-Tate) at a Christmas party in L.A. in 1975. This was before DBase. He's dead now, but wouldn't remember me anyway.

As I was saying, this XBase world is strange. They are heavily into 3rd party add-ons. I've been reading the XBase newsgroups on the internet. Really, I saw someone post a request for a 3rd party library for converting EBCDIC to ASCII. EBCDIC is just the old IBM mainframe character encoding, identical in principle and function to ASCII, just a different mapping of 7 or 8 bit codes to characters. I hope this request is as funny to you as it is to me! Who would consider buying a 3rd party library for this? Any Forth or assembler (and even C, I hope) programmer, would just write his own very short conversion routine. Wow! Talk about how complexity leads to your things needing things. (Does anyone out there have a 3rd party library for incrementing an integer?)

I bring up all this about database programming and its culture, because I have recently taken on the responsibility for maintaining a commercial accounts receivable package written in Clipper. It is sold primarily to doctors offices. (The hot Pentium was in one such office.) Will I rewrite the whole application in Forth? I may. Meanwhile, I must maintain it, enhance it, and live with it in Clipper. This probably sounds mighty strange for someone who writes Forth code for driving stepper motors. On the other hand, I do have an accounting background, having studied accounting in college in my foolish youth and then working in the field some.

If any of you are into the XBase world, please write me with your opinions, suggestions, recommendations. It seems to me that XBase has two main components. One is handling data. This seems like the least troublesome part of building an application. The other component is the user interface. This seems like the most troublesome part. Do the XBase languages really offer anything?

FORCE

FORCE is an XBase version sold by Software Evolution Corporation (contact Herb Curlee at hcurlee@richmond.infi.net, orders at 713-288-8728). It claims to be faster than Clipper and to produce smaller executables. I've rather liked the info I got about it from their ftp site at ftp.vaxxine.com in /pub/msdos/force. With a competing product (almost anything qualifies) I think you can get a copy for around $100. This is considerably cheaper than Clipper. Some posters on the net have cautioned that it might not completely replace Clipper, depending on exactly what you want it to do. Others rave about how they prefer the smaller size and greater speed. I'm tempted to get it to play with, but I'll probably put that off until I've investigated Delphi.

Delphi

This is getting even more complex with the new application generator/memory hogs. For example, Clipper's owner Computer Associates has just come out with VO (Visual Objects), not an upgrade to Clipper, but a whole new thing. It requires Windows and 16 megabytes and I bet it and its applications are bloated and slow. Borland, of course has DBase for Windows, but also has the new Pascal-based Delphi development environment that has been getting good reviews in Jeff Duntemann's *PC Techniques* magazine. In the interests of research, I have recently bought a copy of Delphi. "Rapid Application Development" they say. We'll see. I haven't installed it, but I'll let you know what I find out.

Books

I saw and enjoyed a copy of *The Secret Guide to Computers* in the early 1980s and didn't think about it again until recently. I saw an ad and wrote for the 19th edition a few months ago. I love it! It cost $15 including shipping. It is a nicely put together 639 page paperback. How can anyone sell it for that? I think it must be like hardware stores that used to give away yardsticks. The $15 barely covers or almost covers his costs, and it's so delightful a book that its readers tell everyone, and that keeps the author in speaking engagements. The 20th edition has just come out. Anyway, don't buy a copy! Instead, BUY FOUR COPIES! You see, the author gives volume discounts: $15 for one copy, $12 each for two copies, $9 each for 4 or more copies. Really, buy at least 4 copies. You are going to want to give some away to friends. Mail a check made out to The Secret Guide to Computers to 22 Ashland St., Somerville, MA 02144-3202. If you live in Massachusetts add 5% sales tax. Shipping is free. He offers a money back guarantee, so go ahead and order all four. This is a great book for info on everything about old and modern computers. It's worth the price for either the humor alone or for the technical information, but you get both.

Old Projects

The conversion project I complained about last time still isn't finished, but I've made a lot of progress. Maybe next time I write, it will be completely finished? (I can live in my own little dream world if I want to.) As part of it, to run the new

version of Watcom, I upgraded to a 40MHz '486 motherboard from the 16MHz '386SX motherboard I was using. I got it from Dream Tech at 800-237-3263 or 510-353-1800. I've been fairly happy with them so far. I like the extra speed, which I wouldn't need if I wasn't using Watcom.

MISC and P21

Thinking I might find some spare time, I bought two P21 chips. These are the new Forth chips designed by Chuck Moore and made and sold by C.H.Ting. $25 each unless you are a FIG member, then $20 each. Poor documentation, etc. Apparently very sensitive to exact power supply voltage levels. Perhaps poor video in some/all/many cases? But, the chip is supposed to run at 60 or 80 MHz with built-in NTSC video (for connecting directly to a TV set). It ought to be fun to wire one of these little devils up, and I hope to do so when I have the time. These are "proof of concept" chips and suffer from being constrained to a 40-pin package and a 1.5 micron CMOS process. I applaud these efforts and want to build my own little portable computer, perhaps based on one of these chips. If you want to buy one (or a kit) you can contact C.H. Ting at tingch@ccmail.apldbio.com. Jeff Fox (jfox@netcom.com) is having Chuck build a custom variant of it called the F21. This looks even more exciting, especially if the price is right. I understand a prototype run of F21 chips is scheduled for late May, 1995. An article on the P21 appears in the March/April 1995 *Forth Dimensions* (from FIG).

New Pygmy Not Ready Yet

I am planning to release Pygmy version 1.5 sometime. It isn't ready yet. Progress has ground to a halt because of my other projects. I have conflicting desires: more features versus smaller size and to what degree it should co-operate with the new ANS Forth standard. I expect it to remain a 16-bit Forth. If I ever release its 32-bit big brother, I may need to call it something other than Pygmy so I can charge money for it.

Cheap XTs

Sun Remarketing, 800-821-3221, 801-755-3360, sales@sunrem.com, is offering a PCXT for $29 (plus $38 shipping and $67 for a monitor). They are also offering a '386 or '486 but they don't seem to know which it is. Oh, well, they really specialize in Macs rather than PCs. You might find an XT even cheaper at a garage sale (and not have to pay the shipping). B.G. Micro at 800-276-2206 or 214-271-5546 offers an XT motherboard with 640K RAM for $15.95. By the time you've paid for a case, power supply, keyboard, disk drives, monitor, wouldn't you rather have spent a little more to get a '386 or '486 motherboard instead? Even '386 machines are passe. And, look at the ads for hard disks. I don't think you can find a 120 Mbyte or smaller drive. The smallest in Dream Tech's ad is 420 Mbytes ($189). Gigabyte drives are a little over $300. Now, if RAM prices would only drop. I'm about ready to have, but not to pay for, a CD-R drive. That is a compact disk drive that lets you write onto blank disks, i.e. you can make your own CDs. I keep hearing that prices will be about $1000, but I gather that right now you will really pay about $2000 for the CD-R drive. The blanks can be had for around $10 each or a little less.

Email

If you don't have email access, I think you ought to get it. Essentially all the on-line services (GEnie, AOL, Delphi, Compuserv, Prodigy, etc.) offer links to the internet and thus email access around the world. Another choice might be a local BBS (free or fee) which has such links. My own email addresses have simplified slightly to fs07675@swt.edu (fast) or f.sergeant@GEnie.com (permanent?). If I lived in Austin or San Antonio instead of about half way between them, I might get a commercial account with from a local internet service provider. This would give me a little extra independence from the whims of SWT. The spring semester has just ended. I'm enrolled again in the fall, but not for this summer. Will they cancel my account? Who knows. If I had a commercial internet account, I would use that as my permanent email address. I might get such an account even though it would be a long distance phone call. When I do have my school account, I could telnet to the commercial account from my SWT account to send and receive email, saving the long distance charges. I've become addicted to reading the Usenet (internet) newsgroups, especially comp.lang.forth. I can't pay long distance charges while I browse the newsgroups hour after hour (and I wonder why I'm not finished with my programming projects yet!). While I work all this out, I'll keep GEnie as my permanent email address. It looks like they may be offering full internet access eventually anyway. If you ordinarily talk long distance to one or a few friends, both/all of you switching to email might pay for itself right away. Besides, receiving a phone call ("did I wake you up?") is not always convenient, but receiving email is. I've had recommendations about MCImail but haven't gotten around to checking them out yet. I have an impression that they have an email to U.S. Postal Service mail gateway. If one wanted to isolate himself from dealing with the U.S. Postal Service except when physical items needed to be transferred (and then use FedX or UPS?), MCImail might allow him to handle his correspondence entirely by email on his end, regardless of whether his correspondents had email access.

Miscellaneous

As usual, lots of things remain undone. They just have to wait their turn. I am glad to hear from you. See my email addresses above.

END

Regular Feature

68xx/68xxx Support

C & Assembly

Small System Support

By Ronald W. Anderson

Time to Improve

I note that I've gotten quite far ahead with columns, so I can really take my time on this one. I like to be in this situation and not rushed to get something off in a hurry. That way I have time to let the text languish on my hard drive for a month or two and then go back and read it. I am often a bit embarrassed at how awkward a phrase might be or how many times I've used the same word or phrase. Reading through some previous columns I find that I have frequently said "One more thing ought to be mentioned about ...". That is usually followed by three or four more things. When I have time I can find these and remove the extra verbage. A favorite quote of mine is one attributed to Blaise Pascal: "If I had had more time this would have been a shorter letter". Conciseness comes with pruning and eliminating redundant thoughts. With that, let's get started.

C Tutorial #2

This time let's first have our second installment of C programming tutorial. In our first session we presented a "first program" and discussed the various parts of a C program. This time we will look at some of the rules of C. One of the things we discussed was variable names. We went from the single letter variable names of BASIC to the descriptive names of C without saying much about it. The older BASIC interpreters generally let you use a letter A through Z for a variable name, and most allowed each letter with a 1 through 9 appended, as in Z3. Some let you use two letters as in AN for ANgle and AM for AMount. Depending on the problem that the program had to solve, such variable names were at best cryptic. Meaningful variable names do a great deal to make a program understandable by someone else or by the author after some time has gone by.

In C and most other high level languages both variables and subroutines (called functions in C) can have descriptive names. The ANSI C standard allows up to 32 characters in a name. C programs (by tradition) are written in lower case. The C compiler distinguishes between upper and lower case (this is a compiler option in some compilers), so that names with different capitalization are treated as different by the compiler. It is not a good idea to take advantage of this feature because it is too easy to forget to capitalize a name and therefore use the wrong variable. When such a mistake is made it is hard to find because we forget that the capitalization makes a difference to the compiler. Also a missed capital letter is hard to spot. Quickly, are the following the same or different: DayOfTheWeek — DayOftheWeek. Of course, the second one has a lower case "t" where the first has an upper case. When these are a couple of pages appart it is very hard to spot the difference.

Now for the rules. Variable names must start with a letter. Succeeding symbols may be letters, numbers or an underline character. Variable names contain no spaces. Variable names and function names are called "identifiers". Identifiers should be meaningful. In our example we defined a string variable as an array of characters that I called "message[]". I could just as easily have called it "dave[]" or "xyz[]". The compiler wouldn't care at all, but the goal in naming variables is to have the name mean something to someone else who is reading the program. Underlines can be used in a name to separate the words. Some variable name examples are:

```
pay_rate
day_of_the_week
index
name_6
DayOfTheWeek    // works but not a good idea
```

An identifier can actually start with an underline, but the compiler uses that for some internal names and it is best not to do it within a user program. ANSI C doesn't allow it in all cases so just let the compiler reserve the starting of a name with an underline. Everything we've said about variable names also applies to function names.

One of the reasons that C programmers generally use lower case for variable and function names is that there is a way to define constants in C, and generally those are defined as upper case. That is a clue to someone reading the program that a name is a defined constant. Constants are defined using a compiler directive. We talked about those but didn't call them that. Compiler directives start with a number sign "#" (or pound sign or sharp if you are a musician). We used a compiler directive to include the stdio.h file in our program:

```
#include <stdio.h>
```

We can use a similar directive to define a constant that we can use anywhere in the program:

```
#define PI 3.14159
#define MAXSIZE 25
```

NOTE SPECIFICALLY THAT AN EQUAL SIGN IS NOT

USED AND THERE IS NO SEMICOLON AT THE END OF THE DEFINITION. These "defines" are handled by the C compiler's first pass in which it looks for the #define keyword and then it (for example) finds "PI" wherever it occurs in the program and replaces it with "3.14159". If you forget and use a semicolon, PI will be replaced with "3.14159;". The result will usually be an error message, but it could possibly get by the compiler and become a hard to find bug.

The first compiler pass actually changes the source code and makes these substitutions before the compilation starts. This first pass is called the "Preprocessor" since it proceses the text file before any compilation is done. Sometimes this pass is called the Macro Preprocessor pass for reasons we will see later. There are other compiler directives, specifically those used for "conditional compilation". There may be a section of code that you want to compile only for debug mode and not when you are all done. Sometimes it is nice to leave the debug sections in place for a while so they can be turned on when another bug is found or the program is changed. These directives allow you to turn the compilation of sections of the program on or off by changing one line of the program, a #define directive. When the condition for compilation is turned off the code is not generated. That is, the object code doesn't carry along code that is not used. The only penalty you pay is the larger source code file. It works like this:

```
Early in the program:
#define DEBUG

later:
#ifdef DEBUG
debug code goes here  print a variable name and its
value
etc.
#endif
```

If you don't want the debug code you simply comment out the "#define DEBUG" line. You can also use #ifndef to include code if something is NOT defined.

Assignment Statements

A "statement" in C is an instruction that ends with a semicolon or a group of instructions or statements enclosed by curly braces {}.

```
count=3;
value = 17 * sin(PI/3);
putchar('A');
{
      count=3;
      limit=22;
      value=34;
}
```

The last group of statements enclosed in the curly braces are treated as one statement. It is called a "compound statement". All of the above statements containing an equal sign are "assignment statements". An assignment statement must have a variable name on the left of the equal sign. That is, a statement like:

(a+b) = c; is invalid. You can't assign a value to what we call an "expression". It must be assigned to a variable.

An expression is a combination of variables and constants or literal values that contains "operators". The following is an expression:

```
(a+b) * (c+d)/sqr(1000);
```

Notice that an expression can contain a function. sqr(1000) calculates the square root of 1000. sqr() is a standard library function. In C, something that can properly be used to the left of the equal sign in an assignment is called an "lvalue". Something that can be used to the right of the equal sign is called an rvalue. We can say for certain that variables can be either as in: a = b;. Expressions are always rvalues. You will see these terms in most books about C programming.

One of the things that makes a C program look cryptic is the very large variety of assignment operators. Most languages just have "=". C has a number of them that shortcut both the writing of the program and the execution of it:

```
The Hard Way              The Shortcut

x = x + 1;                 x++;
x = x - 1;                 x--;
x = x + 12;                x += 12; no space +=
x = x / 3;                 x /= 3;       "
x = x * y                  x *= y;       "
```

This may not look much like a shortcut but consider a long variable name for an array element:

```
long_variable_name[17] = long_variable_name[17]+ 3;
long_variable_name[17] +=3;
```

Not only does this save the programmer some time, it causes the compiler to generate less code! It actually runs faster. Basically the short cut says "add 3 to the value of the variable" whereas the long way says "get the value of the variable, add 3 to it and assign the result to the variable".

There are even more shortcut operations. Sometimes it is advantageous to "shift" an integer or a character by one bit position We won't go into all the possible uses at this point, but one thing that can be done is a quick divide or multiply by 2 (or 4 or 8, etc.):

```
a = a >> 1;          a >>= 1;  divides a by 2
a = a << 1;          a <<= 1;  multiplies a by 2.
a = a << 3;          a <<= 3;  multiplies a by 8.
```

Dividing an integer by another does no rounding whatever. 2/3 yields 0 since it is less than 1. 3/2 yields 1. Shifting a variable yields the same results as the divide operator:

```
0000 1001 = 9    9 >> 1     yields 0000 0100    = 4
0000 1001        9 << 1     yields 0001 0010    = 18
```

Next time we'll talk about bitwise AND and OR functions and we will get into the relational operators, i.e. comparisons.

Assembler ala 6809

I am probably moving too fast for those of you with no good reference work on the 6809 instruction set. The TSC assembler manual contains a summary of the instruction set and addressing modes. Mine has that information on pages 19-22. (I did say brief). If you read that, you will see that we haven't come near using all the registers or addressing modes. Don't spend too much time puzzling over the program counter relative mode. We'll discuss that at length when we get into position independent code.

Another good reference would be the accordion folded card that came with these systems. It is an instruction summary card (printed in black and purple). There were several good books on programming the 6809, but surely they are all out of print by now. If you have or can find one, so much the better. If enough of you write to "complain", I'll publish an instruction set summary as part of this column.

At this point I have not yet received any feedback (even from Bill Kibler) regarding the Assembler lessons. I am pleased with them and with the gradual development of more complex programs, but I am not you, and I eagerly await some feedback from the first lesson or two. This time I will continue where we stopped last time. I promised to do a "list" program, one in which we specify a filename on the command line, and the program lists the contents of the specified file to the terminal. Of course FLEX has a LIST utility, but let's write one just to show that it is not all that hard to do.

To do this we need to get into the operating system again and look at how we can read command line parameters from FLEX into our program so we can open the named file. First, I'll present the program, which is just an extension of the one presented last time that could read its own source code. What we have added is a section of code to read the command line parameter and insert it into the File Control Block as a filename. After that the program is the same as SELFDUMP presented last time.

```
* PROGRAM TO LIST A FILE TO THE TERMINAL

NAM LIST

* FLEX SYSTEM EQUATES
PUTCHR  EQU $CD18
WARMS   EQU $CD03
FMS     EQU $D406
FMSCLS  EQU $D403
NXTCH   EQU $CD27       RETURNS NEXT CHARACTER ON THECOMMAND LINE
SETEXT  EQU $CD33       SUPPLIES DEFAULT EXTENSION SET BY CODE IN ACCA

CR      EQU $0D         CARRIAGE RETURN
LF      EQU $0A         LINEFEED
SP      EQU $20         SPACE
PER     EQU $2E         PERIOD
FOPENR  EQU $01         OPEN FOR READ
FCLOSE  EQU $04         CLOSE FILE
EOF     EQU $08         END OF FILE
* CODE ADDED TO GET FILENAME FROM COMMAND LINE

        ORG $C100       THIS IS A FLEX UTILITY PUT IN UCS SPACE
START   LDX #FCB        POINT AT FCB
        LEAX 4,X        SET POINTER TO FILENAME LOCATION IN FCB
INLOOP  JSR NXTCH       GET COMMAND LINE PARAMETER
        CMPA #SP        IS IT A SPACE?
        BEQ INLOOP      IF SO SKIP IT
        CMPA #PER       IS IT A PERIOD?
        BEQ EXTNSN      IF SO GET THE EXTENSION
        CMPA #CR
        BEQ FINISH      CASE OF NO EXTENSION
        STA ,X+         STORE AND INCREMENT
        BRA INLOOP
EXTNSN  LDX #FCB
        LEAX 12,X
EXLOOP  JSR NXTCH       SKIP PERIOD
        BCS FIN         (NXTCH RETURNS CARRY SET ON NON ALPHA)
        STA ,X+         STORE AND INCREMENT
        BRA EXTNSN
FINISH  LDX #FCB
        LDA #1
        JSR SETEXT      SETS EXTN TO .TXT IF NONE SPECIFIED
* END OF NEW CODE

        LDA #FOPENR
        STA 0,X
        JSR FMS
        BNE ERROR       FMS SETS ZERO FLAG FALSE ON ERROR
LOOP    JSR FMS         READ A CHARACTER
        BNE ERROR
        JSR PUTCHR      WRITE IT TO SCREEN
        CMPA #CR
        BNE LOOP        IF NOT, OK
        LDA #LF         ELSE WRITE LF ALSO
        JSR PUTCHR
        BRA LOOP        GO AROUND AGAIN

ERROR   LDB 1,X
        CMPB #EOF       TEST FOR END OF FILE
        BEQ DONE
        JSR RPTERR      TELL USER WHICH ERROR - X POINTING AT FCB
        JSR FMSCLS      CLEAN UP BY CLOSING ALL OPEN FILES ON ERROR
        JMP WARMS
DONE    LDA #FCLOSE     BRANCH HERE ON EOF
        STA 0,X         CLOSE THE FILE
        JSR FMS
        JMP WARMS

FCB     FCB 0,0,0,1     DRIVE 1
        FCB 0,0,0,0,0,0,0,0,0,0,0 ELEVEN ZEROS FOR FILENAME AREA
        RMB 305         THE REST OF THE FCB AREA OF 320 BYTES
        END START
```

We start the new section of code by getting the location of FCB(4) in X. We do this by setting X to the FCB address and then incrementing it by 4. The instruction LEAX 4,X increments X by 4. LEAX means "Load Effective Address X". We will describe it further and use it more when we begin to write programs in position independent code. In the context here it simply means to add 4 to the current value of X. The FLEX routine NXTCH gets the next character from the command line. When you run this program, supposing you want to list the text file LIST.TXT, you type the command line:

```
1.LIST.BIN LIST.TXT
```

The portion "LIST.TXT" is the command line parameter, in this case the name of the file we want to list to our screen. FLEX allows you to use a space or a comma to separate parameters. This program will expect a space.

Back to NXTCH. It gets a character from the command line. We test for space ($20) and skip it if detected by going back to get another character. If it wasn't a space we test for a period ($2E). If we detect a period we are done with the filename and need to go handle the extension. If it is not a period we check for a CR. If we find one, there was no extension so we branch to our label DONE.

If we got past all these tests we have a valid character of the filename so we put it in the FCB with STA ,X+. This stores the

character at the address pointed at by X and increments X by one. (This ought to ring bells, since it is just like the "X++" sequence we were talking about above in the discussion of C programming). Then we go around the loop again. When we find the period we branch to the label EXTNSN. You will see later that I preloaded the FCB filename and extension area with zeros, so if the filename is shorter than 8 characters we already have the padding zeros in place. Now we must set the X pointer to point at the extension locations which start at FCB(12), so we load X with the FCB address and do an LEAX 12,X. Actually, we could let the assembler do some arithmetic for us and LDX #FCB+12. We could also have used this for the FCB+4 above. Now we store valid alpha characters until we get to the end of the parameter line. The extension can be one, two, or three characters long. Again we have pre-padded with zeros so it doesn't matter.

I decided to have this program do one more thing in order to illustrate the use of SETEXT. If the command line had a filename without an extension (i.e. no period) a default one can be supplied by the operating system. LDA #1 sets the code to the .TXT extension. (See the Advanced Programmer's Guide SETEXT explanation for a table of codes for default extensions). JSR SETEXT now will fill in a missing extension. If you supplied any extension at all, it will be left alone. The only other thing to note is that my definition of the File control block has changed from last time. We set the first four bytes to 0,0,0,1 where 1 is the drive number for the working drive. Then we clear the next 11 bytes to zero, so that when we load the filename into it we don't have to bother to count characters and fill the empty spots with zeros.

Type this program in and try running it. If you decide to keep your very own list program copy it to your system disk:

```
COPY 1.LIST.BIN 0.MYLIST.CMD
```

Dont simply copy it as LIST.CMD because you already have the FLEX LIST.CMD on your system disk, and the FLEX supplied one is smarter than this one. The FLEX one has an option for adding line numbers when it lists a file. If you try to list a long file you might notice that the listing goes whizzing by on the screen and doesn't stop even if you have the TTYSET PAUSE parameter set to ON and you have a number of lines set to 22 or 23. That is because the pause mechanism of FLEX only counts lines if they are terminated by a call to the PCRLF system subroutine. Let's change our program to do that:

First the equate PCRLF EQU $CD24 must be added. Then, rather than test for CR and add LF we do the following starting at the label LOOP in the program:

```
LOOP    JSR FMS         READ A CHARACTER
        BNE ERROR
        CMPA #CR        IS IT A CR
        BNE CONTIN
        JSR PCRLF
        BRA LOOP        WITHOUT OUTPUTTING THE CR
CONTIN JSR PUTCHR
        BRA LOOP
ERROR .....
```

Now when the progam encounters a CR it will call PCRLF which will count the line, and the pause feature will work properly so you can list your file one screen at a time. There's one other thing you could do with this program. You can make it default to the working drive, (i.e. whichever drive you have made the working drive) by using the FLEX variable I'll call WRKDRV at $CC0C. Flip back a few pages in the advanced programmer's guide to the list of FLEX parameter locations and you will find it. All you have to do is the equate: WRKDRV EQU $CC0C. Then when you have X pointing at the FCB add two instructions LDA WRKDRV, and STA 3,X. The best place to insert these is just after the line labeled FINISH. this will overwrite the hard coded value 1 that was put in the FCB. Of course if you always use drive 1 as your working drive this is an unnecessary exercise.

It occurs to me that maybe you are wondering what that extra accumulator ACCB is good for. so far we haven't used it very much. You can use it to advantage when ACCA is being used for something else. Suppose for example that you want to output exactly 8 spaces. You might use B for a counter:

```
        LDB #8
        LDA #$20   SPACE CHARACTER
LOOP    JSR PUTCHR
        DECB
        BNE LOOP
```

Of course we have discussed the D accumulator, the concatenation of A and B where B is the low order byte and A is the high order byte of the D register's 16 bit value. I note that we haven't used the Y index register or the U user stack pointer either. There are good uses for these and we will get into some examples soon.

Miscellaneous

I am including this simply because it might help someone else find bugs in a PC system. Some time ago I added a CD-ROM drive to my computer, one that I have put together and upgraded from time to time. After a while I noted that I couldn't read the CD-ROM. It would start up and the boot sequence would say it was detected and set up as drive E: I would do a dir E: and get back the drive not ready error message. I finally tracked the problem down to having a power supply that was inadequate for my newer 386 board with 8 megs of memory plus the CD-ROM drive plus a newer and larger hard drive.

I replaced the old 150 watt power supply with a 200 and the CD-ROM problems went away completely. Then my new 420 Meg hard drive began not to start when the computer was turned on. It would sometimes start if I turned the switch off and then back on. Finally it got to the point where it wouldn't start at all. I found that by disconnecting the power connector and plugging it in again, the drive would start and all would work fine until I turned the computer off again. At that point I figured my drive had gotten "sticky". I had that problem a while ago with a 20 meg drive on a 68000 system. Since the drive is barely two months old I phoned Peripheral Technol-

ogy, the supplier, and they very kindly agreed to send me a new one so I could transfer files to it before sending back the defective one.

The new drive arrived and I plugged it in, ready for an all night session copying files, only to note exactly the same problem. It wouldn't start! After all my part swapping (the CD-ROM drive worked fine in another computer) I felt sure I had swapped IDE drive controllers but maybe not since the hard drive started acting up. I swapped controllers with my spare computer (made up of parts left over from upgrading my computer, my wife's computer and my daughter's), and wonder of wonders, the new drive and my old one both worked fine.

Thank you, Fred Brown of Peripheral Technology, for being so kind to get me a new drive in three days! Actually I had requisitioned one of these exact drives a few days ago for the PC in my office, so we called Fred and told him to invoice the company for the drive. I connected it today along with the drive that was already there, copied all my files over to the new drive, reconfigured it as the main drive, and I was all done. Just in case that didn't work, however, I had backed up most of the files that were not "reinstallable". My "old" 170 meg drive (about 15 months old) will go into someone else's system that has outgrown an 80 meg drive.

About now you are wondering how many computers I have. My son had a PC, the best in the family, which grew and was upgraded over several birthdays and Christmases. At that time I was involved in the Peripheral Technology 68000 computer systems.

A couple of years ago I bought an old 6 MHz 286 motherboard for $75 and then added a case with power supply so I could have my own computer. I did some extra work and was paid with a monitor and hard drive. Eventually I got a system together. After a number of upgrades I had a 386-40 system with 8 megs of RAM and the 170 Mbyte hard drive. Last January I added a Mitsumi double-spin CD rom drive.

Meanwhile, my daughter had started with an old PC-XT when she was in college. That got upgraded to a used 286-12 motherboard a couple years ago. The hard drive was upgraded several times as well. The original 10 Mbyte was replaced by an RLL 30, then an MFM 40, a 20 added to that for 60, and later a 68 MFM. Last Christmas I added a 386-40 motherboard to that, and put a system together for my wife with some of the leftovers. Presently she only uses it for playing games when she wants a change of pace, but she has expressed interest in learning to use a word processor for letters etc. My son is in Graphic Arts, and he acquired a Mac, so he gave me back his PC system. We did a fast shuffle. My daughter was upgraded to a 170 meg hard drive and a good VGA monitor. My wife had been using the 286-12, and she acquired the 386-40 board. My daughter had an EGA monitor so that, a 68 meg MFM drive, and the 286-12 ended up in an extra system. I brought that down to work just to have an extra there.

Most recently I upgraded my system to a 486SLC-66/2 (i.e. the processor runs at 66 MHz., but the buss runs at 33. The system includes a co-processor that runs at 33 MHz. The company is using these boards in our product and getting a very good price on them — so good I couldn't resist. The leftover 386-40 went in the extra computer. I figure if I can keep it running, I have a source of known good spare parts. With my recent power supply problems, it lost its working 200 watt supply and I am short a good power supply or maybe a case, which doesn't cost that much more than a power supply. I sold the EGA monitor and controller for about 1/5 of what I paid for them just before VGA took over the world. I felt lucky to get rid of it and a friend who was building a system for his daughter to take to college was glad to have it at a reasonable price.

That isn't the whole story, but you get the picture. When one computer is upgraded, if the removed piece is better than that in one of the others, it is moved to that system etc. down the line. Things that bump off the end of the line stay in my "inventory" to use as test items if someone asks me to repair a computer, or perhaps get sold for present fair value minus to friends who are working on systems for their children "so I can use my computer sometimes". Recently a friend donated an old PC-XT to the Salvation Army. It had a glitch and I repaired a hard disk controller on it and then decided to remove the 20 meg drive and throw in a 40 that was in my excess inventory.

Now I think I'll leave this computer alone for a while and just use it rather than replacing something and causing troubles again.

Sometimes people ask me what brand of PC clone I have. I tell them that I have never bought a complete computer. I built the 6809 system from kits. Ditto for a 68000 system from Peripheral Technology. Quite obviously the PCs that I use are also put together from parts, some ordered by mail and most bought locally at a little computer store that just about matches mail order prices but has a limited choice of items in their inventory.

I must admit that I've made a few dumb mistakes installing hardware that have cost me some of my old components and a couple of repairs on MFM hard drives. I call that educational expenses. If you never try something for fear of ruining materials, you will never know if you could have done it or not. When we first were married (a long time ago) my wife was afraid to try sewing some fancy project for fear of making a mistake and scrapping the fabric she would have to buy. I told her that she could think of an error as educational expense, throw the fabric away, buy some new, and do it right the second time. I'm not saying that a dumb mistake doesn't make me feel stupid, but I think I am over the hump with regard to putting computer parts and pieces together.

TCJ Center Fold

Special Feature

All Users

640K PC/XT Clone

I purchased a 640K board many years ago. This manual came with it, and has no copyrights or manufacturers names. This is typical of clones and allows me to reprint it. I have not edited the text except to get it in WordStar form. Note the bad grammar, spelling, and poor schematics (which I hope you can still read at this point, copied several times before I got them.) Read the I/O descriptions and some of the interface specifications, especially the line where the expansion bus is an 8080 enhanced interface! I found a number of very interesting and revealing specifications that explain why it is so poor an interface. See if you can find them too. BDK

THE SYSTEM BOARD

The PC/XT system board fits horizontally in the base of the system unit and is approximately 8 1/2 x 12 inches. It is a double sided P.C.B., DC power and a signal from the power supply enter the board through two six-pin connectors. Other connectors on the board are for attaching the keyboard and speaker. Eight 62-pin card edge-sockets are also mounted on the board. The I/O channel is bussed across these eight I/O slots.

A "Dual-in-Line Package (DIP) switch (SW1) (one eight-switch pack) is mounted on the board and can be read under program control. The DIP switch provides the system software with information about the installed options, how much storage the system board has, what type of the display adapter is installed, what operation modes are desired when power is switched on (color or black-and-white, 80- or 40-character lines), and the number of diskette drive attached.

The system board consists of five functional areas: the processor subsystem and its support elements, the Read-Only Memory (ROM) subsystem, the Read/Write (R/W) Memory subsystem, integrated I/O adapters, and the I/O channel. All are described in this section.

The heart of the PC/XT system board is the Intel 8088 microprocessor. This processor is an 8-bit external bus version of Intel's 16-bit 8086 processor, and it's software-compatible with the 8086. Thus, the 8088 supports 16-bit operations, including multiply and divide, and supports 20 bits addressing (1 megabyte of storage). It also operates in a maximum mode, so a coprocessor can be added as a feature. The processor operates at 4.77 MHz. This frequency, which is derived from a 14.318 MHz crystal, is divided by 3 for the processor clock, and by 4 to obtain the 3.58 MHz color burst signal required for color televisions.

At the 4.77 MHz clock rate, the 8088 bus cycles are four clocks of 210 ns, or 840 ns. I/O cycles take five 210 ns clocks or 1.05 microseconds.

The processor is supported by a set of high-function devices providing four channels of 20-bit Direct-Memory Access (DMA), three 16-bit timer-counter channels, and eight prioritized interrupt levels.

Three of the four DMA channels are available on the I/O bus and support high speed data transfers between I/O devices and memory without processor intervention. The fourth DMA channel is programmed to refresh the system dynamic memory. This is done by programing a channel of the timer-counter device to periodically request a dummy DMA transfer. This action creates a memory-read cycle, which is available to refresh dynamic storage, both on the system board and in the system expansion slots. All DMA data transfers, except the refresh channel, take five processor clocks of 210 ns, or 1.05,us if the processor ready line is not deactivated. Refresh DMA cycles take four clocks or 840 ns.

The three programmable timer/counters are used by the system as follows: Channel 0 is used as a general purpose timer, providing a constant time base for implementing a time-of-day clock; Channel 1 is used to time and request refresh cycles from the DMA channel; and Channel 2 is used to support the tone generation for the audio speaker. Each channel has a minimum timing resolution of 1.05us.

Of the eight prioritized.levels of interrupt, six are bussed to the system expansion slots for use by features cards. Two levels are used on the system board. Level 0, the highest priority, is attached to Channel 0 of the timer/ counter and provides a periodic interrupt for the time-of-day clock. Level 1 is attached to the keyboard adapter circuits and receives an interrupt for each scan code sent by the keyboard. The Non-Maskable Interrupt (NMI) of the 8088 is used to report memory parity errors.

The system board supports both ROM/EPROM and R/W memory. It has space for 128K x 8 of ROM or EPROM. This ROM contains the power-on self-test, I/O drivers, dot patterns

for 128 characters in graphics mode, and a diskette and has an access time and a cycle time of 250 ns each.

The system board also has from 128K 9 to 256K 9 of R/W memory. A minimum system would have 128K of memory, with module sockets for an additional 128K. Memory greater than the system board has a maximum of 256K, is obtained by adding memory cards in the expansion slots. The memory consists of dynamic 64K 1 chips with an access time of 200 ns and a cycle time of 345 ns. All R/W memory is parity checked.

The system board contains the adapter circuits for attaching the serial interface from the keyboard. These circuits generate an interrupt on to the processor, when a complete scan code is received. The interface can request execution of a diagnostic test in the keyboard.

The keyboard interface is a 5-pin DIN connector on the system board, that extends through the rear panel of the system unit.

The system units has an 2 1/4 inch audio speaker. The speaker's control circuits and driver are on the system board. The speaker connect through a 2-wire interface that attaches to a 3-pin connector on the system board.

The speaker drive circuit could be capable of providing approximately 1/2 watt of power. The control circuits allow the speaker to be driven three different ways: 1) a direct program control register bit may be toggled, to generate a pulse train; 2) the output from Channel 2 of the timer counter, may be programed to generate a waveform to the speaker; 3) the clock input to the timer counter, can be modulated with a program controlled I/O register bit. All three methods may be performed simultaneously.

Section II

I/O Channel

The I/O channel is an extension of the 8080 microprocessor bus. It is, however, demultiplexed, repowered, and enhanced by the addition of interrupts and Direct Memory Access (DMA) functions.

The I/O channel contains an 8 bit, bidirectional data bus, 20 address lines, 6 levels of interrupt, control lines for memory and I/O read or write, clock and timing lines, 3 channels of DMA control lines, memory refresh timing control lines, a channel check line, and power and ground for the adapters. Four voltage levels are provided for I/O card: +5 Vdc, -5 Vdc.'+12 Vdc, and -12dc. These functions are provided in a 62-pin connector with 100-mil card tab spacing.

A "ready" line is available on the I/O channel, to allow operation with slow I/O or memory devices. If the channel's ready line is not activated by an addressed device, all processor-generated memory read and write cycles takes four 210-ns clock or 840-ns/byte. All processor-generated I/O read and write cycles require five clocks for a cycle time of 1.05us/byte. Refresh cycles occur once every 72 clocks (approximately 15us) and require four clocks or approximately 7% of the bus bandwidth.

I/O devices are addressed using I/O mapped address space. The channel is designed so that 768 I/O device addresses are available to the I/O channel cards.

A channel check line exists for reporting error conditions to the processor. Activating this line results in a Non-Maskable Interrupt (NMI) to the 8088 processor. Memory expansion options use this line to report parity error$.

The I/O channel is repowered, to provide sufficient drive, to power all eight (Jl through J8) expansion slots, assuming two Low-Power Sohorttky (LS) loads per slot. The I/O adapters typically use only one load.

Section III

I/O Channel Description

The following is a description of the PC/XT I/O Channel. All lines are TTL-compatible.

Signal I/O Description

OSC, Oscillator: High: speed clock with a 70-ns period (14.31818 MHz). It has a 50% duty cycle.

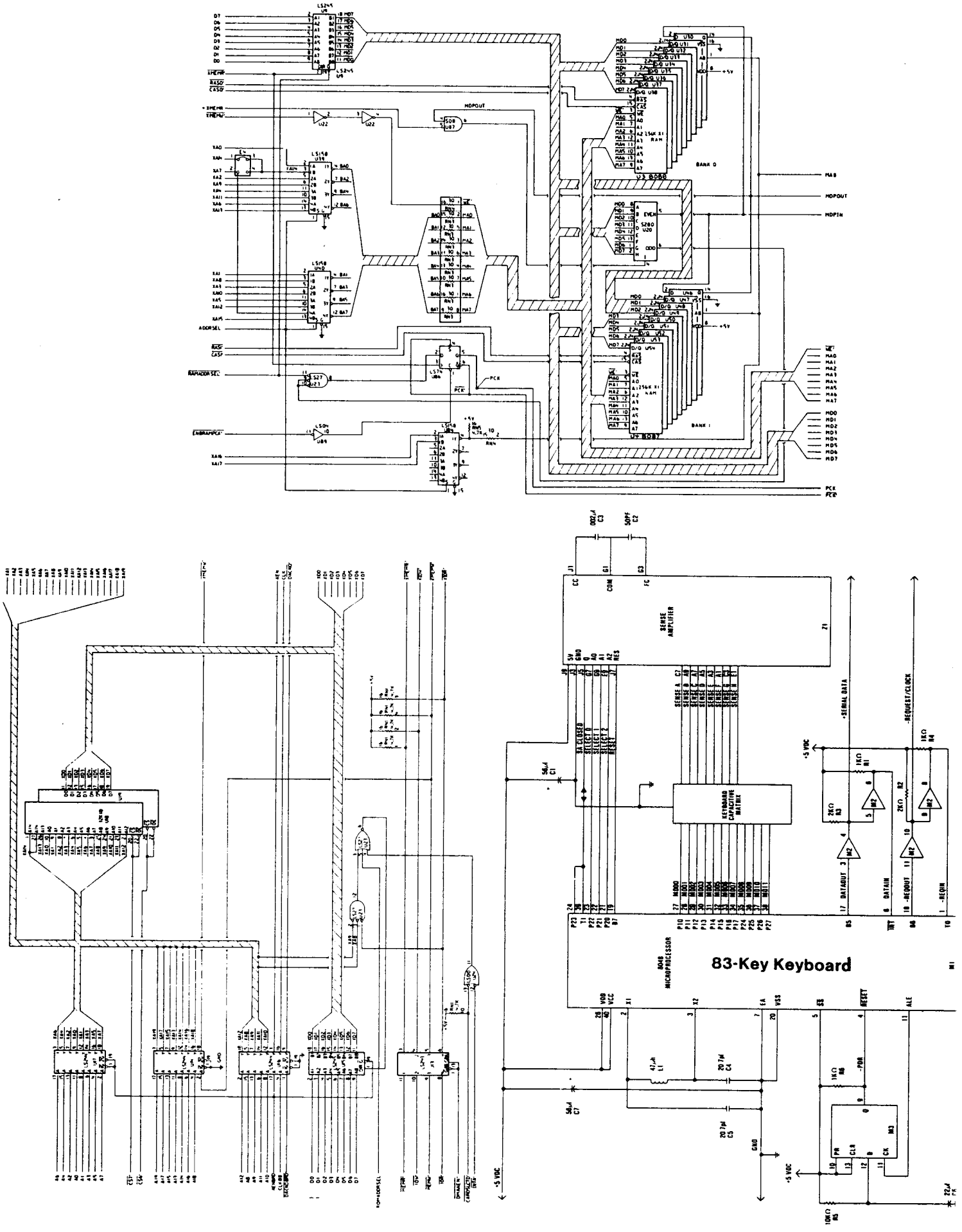
CLK, System Clock: It is divide-by-three of the oscillator and has a period of 210 ns (4.77 MHz). The clock has a 33% duty cycle.

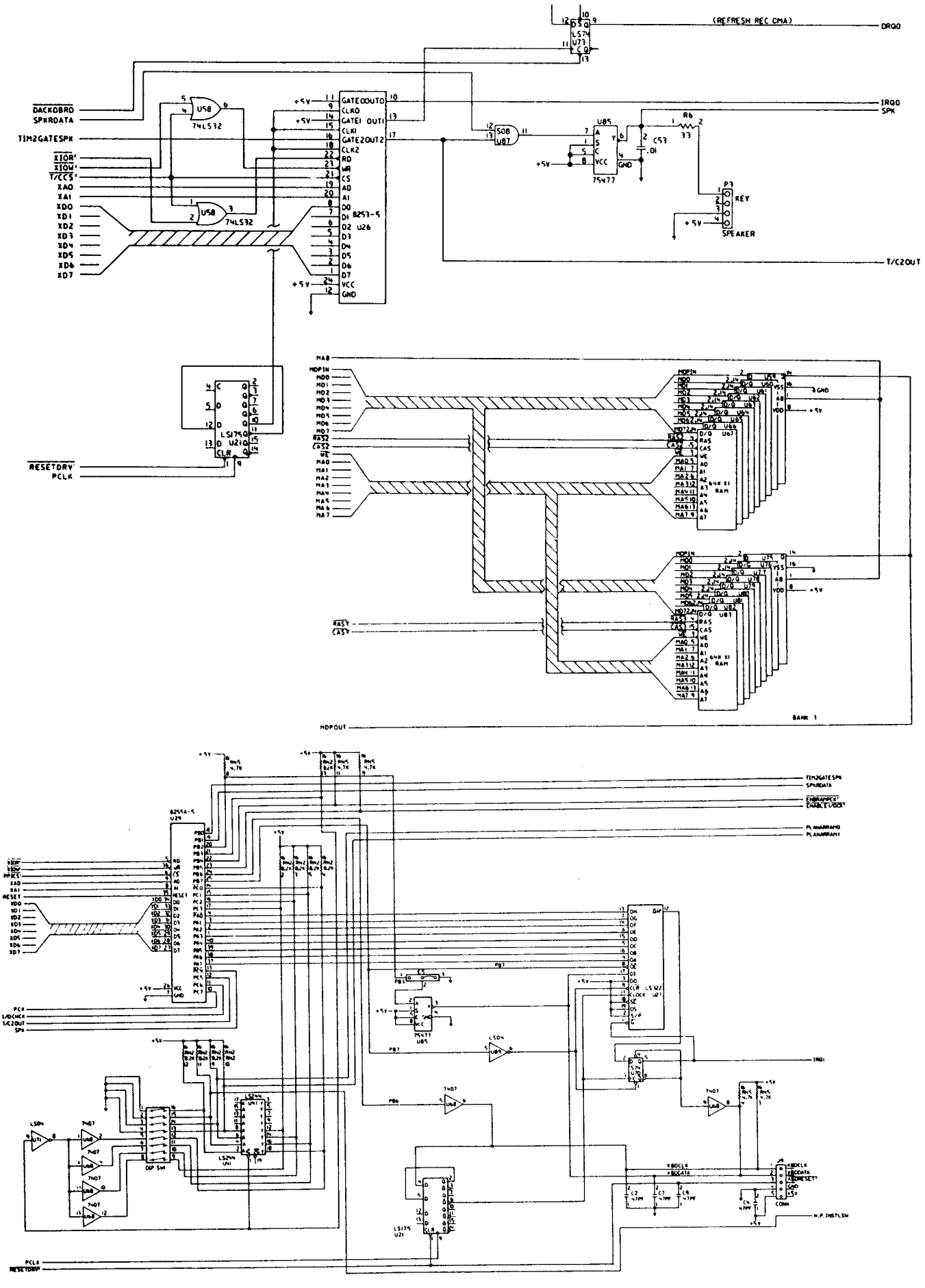
RESET: This line is used to reset or initialize system logic upon power-up or during a low line voltage outage. This signal is synchronized to the failing edge of clock and is active high.

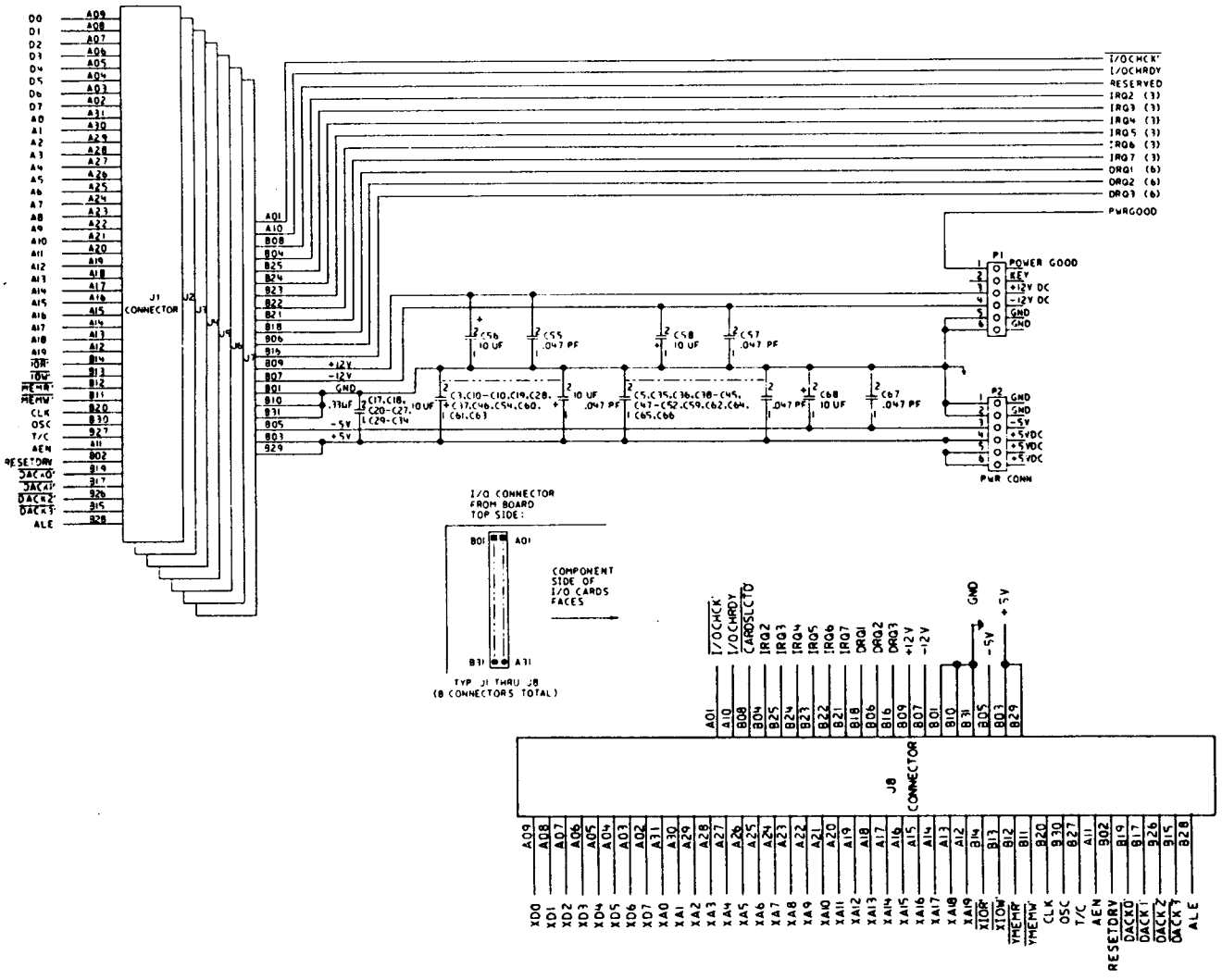
A0-A19, Address Bits 0 to 19: These lines are used to address memory and I/O devices within the system. The 20 address lines allow access of up to 1 megabyte of memory. A0 is the Least Significant Bit (LSB) and A19 is the Most Significant Bit (MSB). These lines are generated by either the processor or DMA controller. They are active high.

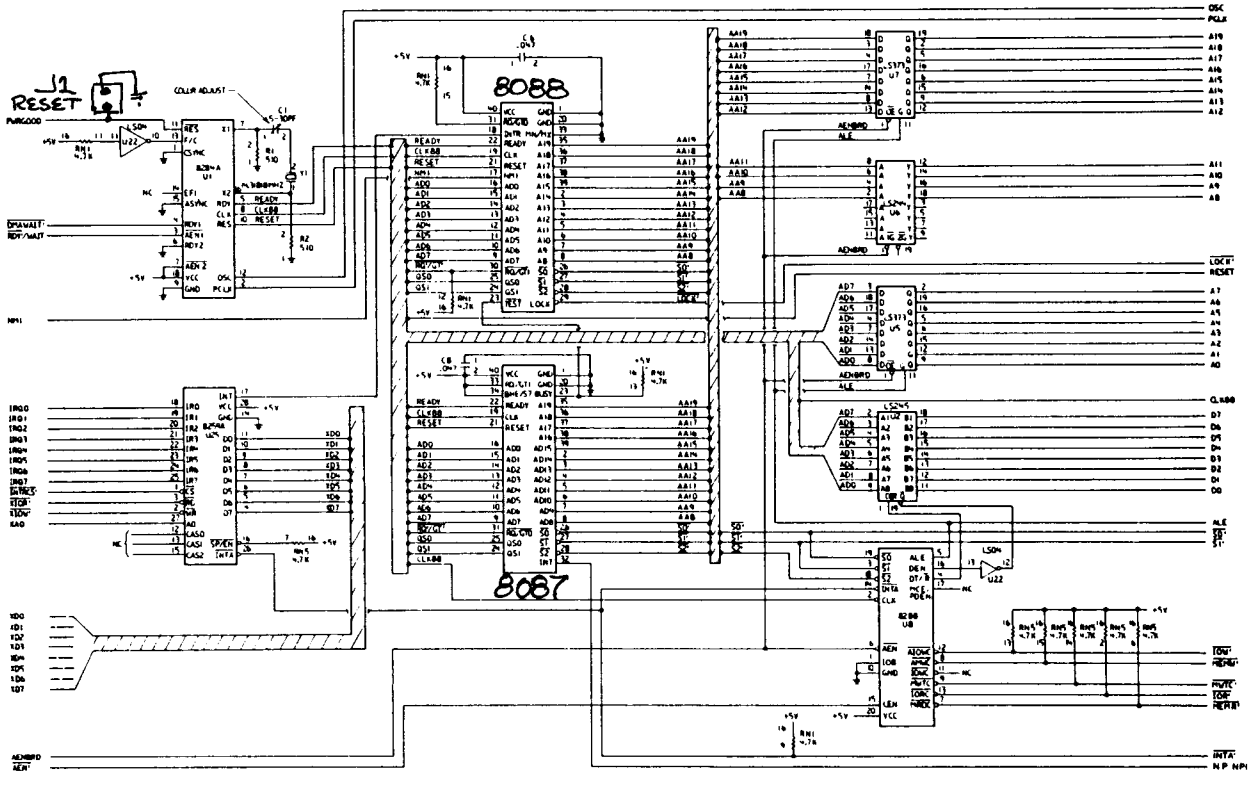
D0-D7, I/O Data Bits 0 to 7: These lines provide data bus bits 0 to 7 for the processor, memory, and I/O devices. D0 is the Least Significant Bit (LSB) and D7 is the Most Significant Bit (MSB). These lines are active high.

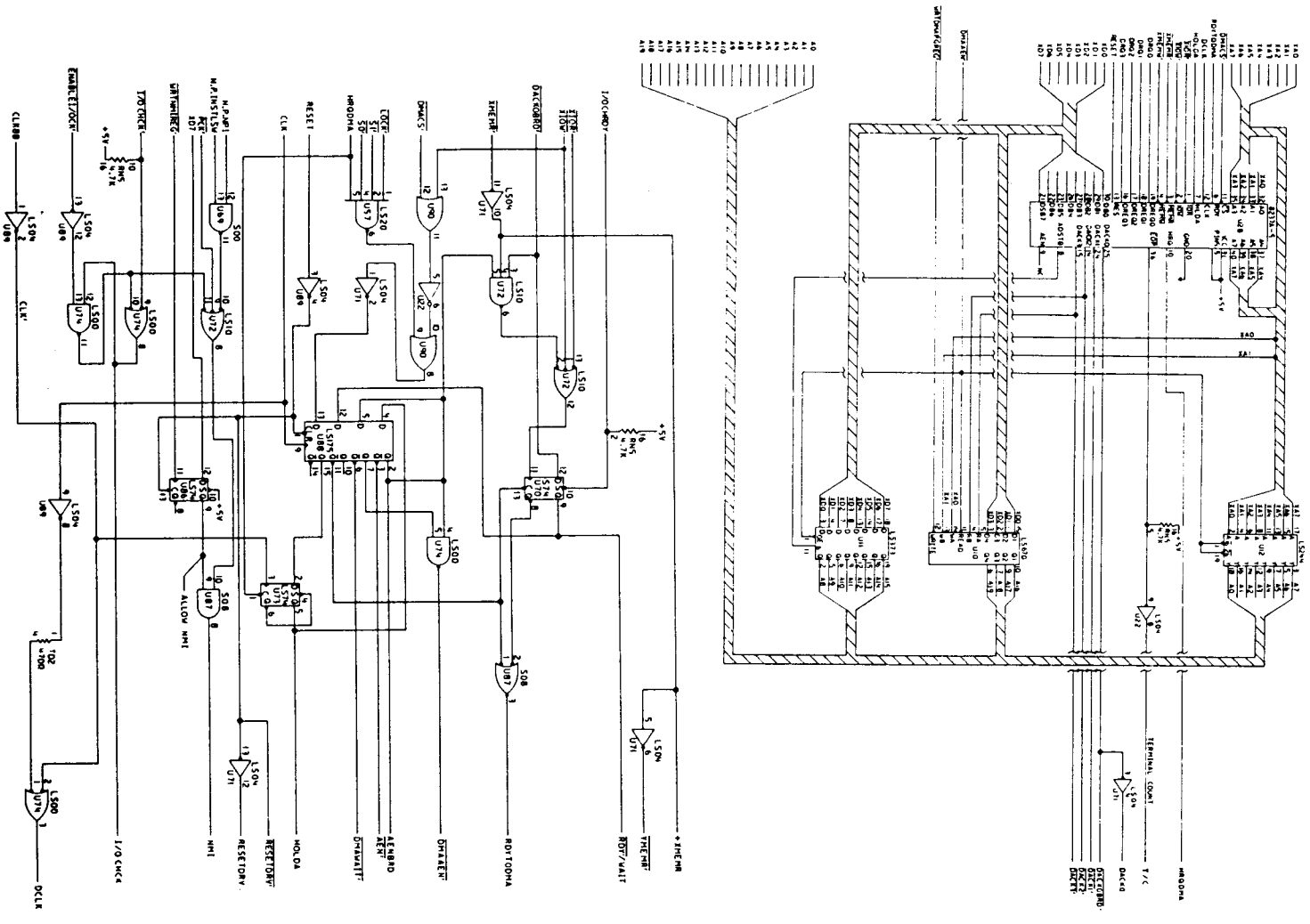
ALE, Address Latch Enable: This line is provided by the 8288 Bus Controller and is used on the system board to latch valid addresses from the processor. It is available to the I/O channel as an indicator of a valid processor address (when used with AEN). Processor addresses are latched with the failing edge of ALE.

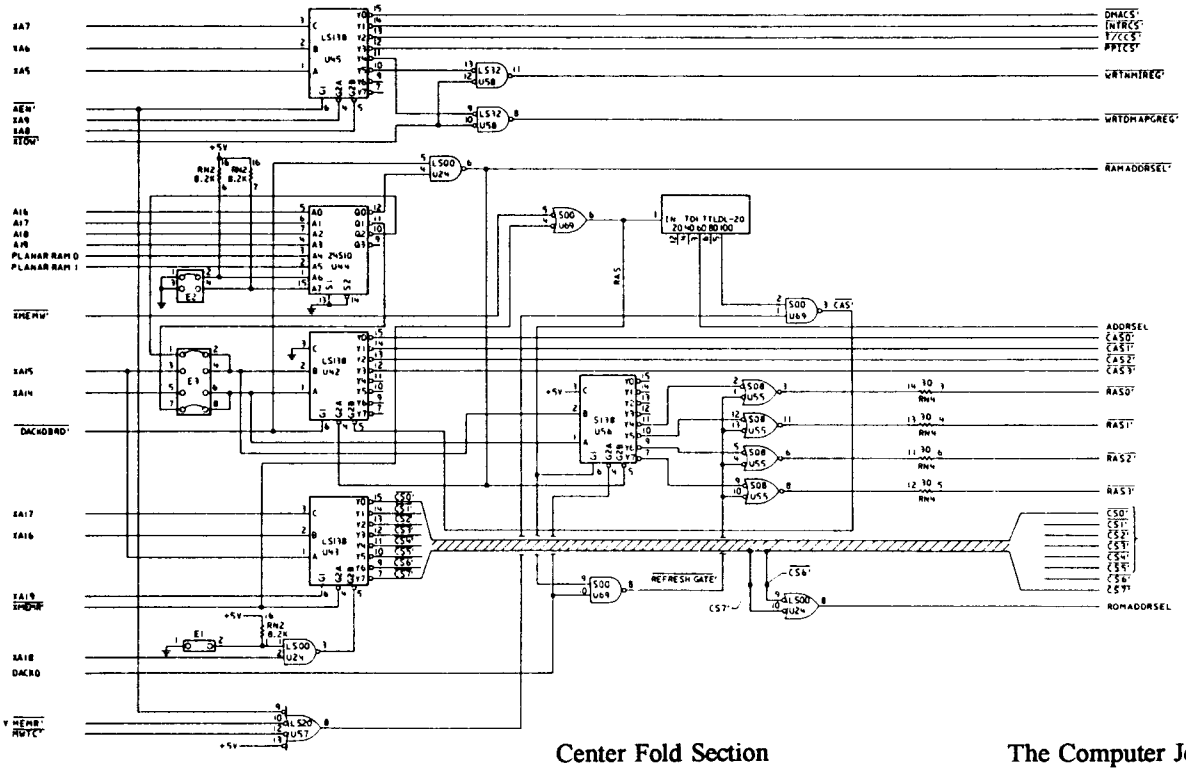

83-Key Keyboard


(REFRESH REC CMA)
DRQ0
IRQ0
SPK
T/C2OUT
DACKOBRD
SPKRDATA
TIM2GATESPK
XIOR'
XIOW'
T/CCS'
XA0
XA1
XD0
XD1
XD2
XD3
XD4
XD5
XD6
XD7
74LS32
8253-5
U26
75477
SPEAKER
KEY
RESETDRV
PCLK
RASY
CASY
MDPOUT
BANK 1
64K X1 RAM
8255A-5
U29
DIP SW
IRQ1
PCLK
RESETDRV


J1 CONNECTOR
I/O CONNECTOR FROM BOARD TOP SIDE:
COMPONENT SIDE OF I/O CARDS FACES
TYP J1 THRU J8 (8 CONNECTORS TOTAL)
P1 POWER GOOD
P2
PWR CONN
J8 CONNECTOR
I/OCHCK
I/OCHRDY
CARDSLCTD
IRQ2
IRQ3
IRQ4
IRQ5
IRQ6
IRQ7
DRQ1
DRQ2
DRQ3
+12V
-12V
GND
-5V
+5V
PWRGOOD
RESERVED
ALE


J1
RESET
8088
8087
8284A
8288
LS245
LS373
LS244
LS138
LS04
LS00
LS32
LS20
LS10
LS08
LS27
LS157
LS153
LS74
LS86
LS02
LS51
LS22
LS37
LS125
LS367
LS368
LS139
LS14
LS175
LS273
LS32
LS00
LS04
LS08
LS10
LS20
LS27
LS74
LS86
LS02
LS51
LS22
LS37
LS125
LS367
LS368
LS139
LS14
LS175
LS273





I/O CH CK, I/O Channel Check: This line provides the processor with parity (error) information on memory or devices in the I/O channel. When this signal is active low, a parity error would be indicated.

I/O CH RDY, I/O Channel Ready: This line, normally high (ready), can be pulled low (not ready) by a memory or I/O device to lengthen I/O or memory cycles. It allows slower devices to attach to the I/O channel with a minimum of difficulty. Any slow device using this line should drive it low immediately upon detecting a valid address and a read or write command. This line should never be held low, longer than 10 clock cycles. Machine cycles (I/O or memory) are extended by an integral number of CLK cycles (210 ns).

IRQ2-IRQ7, Interrupt Request 2 to 7: These lines are used to signal the processor, that a I/O device requires attention. They are prioritized with IRQ2 as the highest priority and IRQ7 as the lowest. An Interrupt Request are generated by raising an IRQ line (low to high) and holding it high, until it was acknowledged by the processor (interrupt service routine).

IOR, I/O Read Command: This command line instructs an I/O device to drive its data onto the data bus. It may be driven by the processor or the DMA controller. This signal is active low.

IOW, I/O Write Command: This command line instructs an I/O device, to read the data on the data bus. It may be driven by the processor or the DMA controller. This signal is active low.

MEMR. Memory Read Command: This command line instructs the memory to drive its data into the data bus. It may be driven by the processor or the DMA controller. This signal is active low.

MEMW. Memory Write Command: This command line instructs the memory to store the data present on the data bus. It may be driven by the processor or the DMA controller. This signal is active low.

DRQ1-DRQ3, DMA Request 1 to 3: These lines are asynchronous channel requests used by peripheral devices to gain DMA service. They are prioritized with DRQ3 being the lowest and DRQ1 being the highest. A request is generated by bringing a DRQ line to an active level (high). A DRQ line must be held high until the corresponding DACK line goes active.

DACK0-3-DMA Acknowledge 0 to 3: These lines are DACK3 used to acknowledge DMA requests (DRQ1-DRQ3) and to refresh system dynamic memory (DACK0). They are active low.

AEN, Address Enable: This line is used to degate the processor and other devices from the I/O channel to allow DMA transfers to take place. When this line is active (high), the DMA controller has control the address bus, data bus, read command lines (memory and I/O), and the write command lines (memory and I/O).

T/C, Terminal Count: This line provides a pulse when the terminal count for any DMA channel is reached. This signal is active high.

CARD SLCTD, Card Selected: This line is activated by cards in expansion slot J8. It signals, "THE SYSTEM BOARD", that the card has been selected and that appropriate drivers on the system board should be directed to either read from, or write to, expansion slot J8. Connectors J1 through J8 are tied together at this pin, but the system board should be driven by an open collector device.

The following voltages are available on the system board I/O channel:

+5 Vdc +-5%, located on 2 connector pins -5 Vdc +-10%, located on 1 connector pin +12 Vdc +-5%, located on 1,,connector pin -12 Vdc +-10%, located on 1 connector pin GND (Ground), located on 3 connector pins

Section IV

Speaker Interface

The sound system has a small, permanent-magnet, 2 1/4 inch speaker. The speaker can be driven from one or two sources:

An 8255A-5 PPI output bit. The address and bit are defined in the "I/O Address Map"

A timer clock channel, the output of which is Programmable within the functions of the 8253-5 timer when using a 1.19-MHz clock input. The timer gate also is controlled by an 8255A-5 PPI output-port bit. Address and bit assignment are in the "I/O Address Map".

The speaker connection is a 4-pin berg connector. See "System Board Component Diagram", earlier in this section, for speaker connection or placement.

Section V

Some Trouble Shooting Points

This section give some hints to experienced engineers and technicians to test and bring up the Super XT board.

1) System Clock: 4.77 MHz at U1 (8284A) PIN 8 and U3 (8088) PIN 10, etc., with 33% duty cycle. If no system clock, check Y1 (14.318MHz) crystal, U1.

2) Power Up Reset: When Power up U3 (8088) PIN 21 should go high then back to low again. If no power up, please check jumper on JP1.

3) Memory Decoding Circuit: When power up, the processor will start execute the program through ROM 7 (BIOS & Self Test). The CS7 - line should see a train low again pulses to fetch instruction from ROM 7. If no pulse on CS7 - on U23 (LS138 PIN &), check the memory decoding circuit and make sure the ROM/EPROM jumpers (W1-W12) are in the right position.

4) RAM Refresh: The RAM is being refresh using the time chip U49 (8253-5 PIN 13 OUT 1) to trigger the chip (8237A-5) DRQ0, and the DMA chip will send out a dummy address to refresh the dynamic RAMs (4164). If no periodic low pulses on RAM chip PIN 4 (RAS-), check the TIMER, DMA, and refreshing circuit.

5) R.T.C. Circuit: A square wave is generated through TIMER/ COUNTER CHIP (8253-5 U49) PIN 10 (OUT 0), and then interrupt the processor through 8259A PIN 17 (INT Signal) for the R. T. C. Timing.

6) Keyboard Circuit: The keyboard data input is through keyboard connector PIN 2 and U22 (LS322) to convert the serial data to parallel and read by 8255A-5 (U37) port A. Interrupt 1 (ITQ 1) is used to inform the processor, that a key had been pressed. If no keyboard response, check U52 (74LS74), U22 (LS322), and U37 (8255A-5).

NOTE: The PC/XT system board will run with the XT ROM set (32Kx8), but will not run with the PC ROM set (8Kx8).

NOTE: Some system boards contain Hardware Reset Jumper (JP1), if use a S.W. (i.e. normal open) to connect Jumper 1, when push s.w. (i.e. short), the system board hardware will be reset.

NOTE: Press and hold the Ctrl (control) and Alt (Alternate) keys and then press the Del (Delete) key, then release all three, this operation is called software reset.

Section VI

The system board Switch setting

The DIP Switch (SW1) located at U20 is used to set the system configuration and specify the amount of memory installed on the system board.

Position	Function
1	Normal operation off
2	Use-for 8087 co-processor
3-4	Amount of memory on system board
5-6	Type of display adapter
7-8	Number of 5 1/4 inch diskette drives

Switch (SW1):
1-OFF (NORMAL OPERATION)
2-ON W/O 8087 co-processor
2-OFF W 8087 co-processor

Memory Switch Settings:
3-OFF 4=ON 128K MEMORY INSTALLED
3-ON 4-OFF 192K MEMORY INSTALLED
3-OFF 4=OFF 256K MEMORY INSTALLED

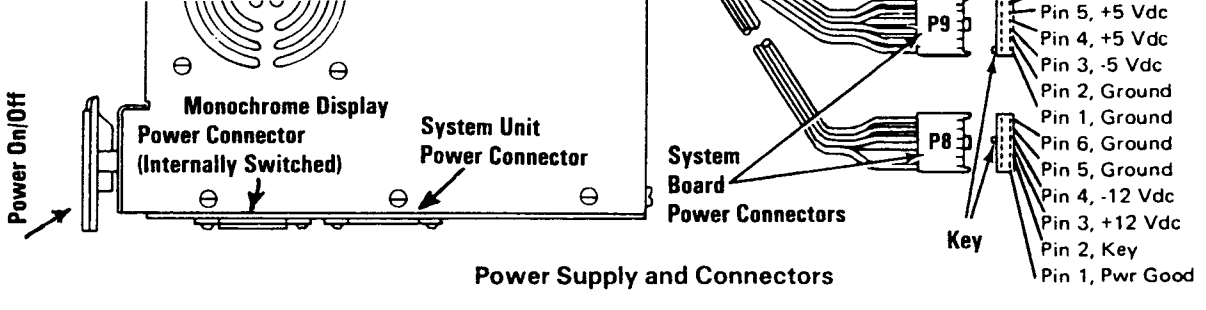
Display Adapter Switch Settings:
5-ON 6-ON NO DISPLAY ADAPTER
6-OFF 6-ON COLOR/GRAPHICS (4OX20 Mode)
5=ON 6-OFF COLOR/GRAPHICS (BOX25 Mode)
5-OFF 6=OFF MONOCHROME DISPLAY ADAPTER OR BOTH

Display Drive Switch Setting:
7=ON 8-ON 1 DRIVE INSTALLED
7=OFF 8=ON 2 DRIVES INSTALLED
7-ON 8-OFF 3 DRIVES INSTALLED
7=OFF 8-OFF 4 DRIVES INSTALLED

Pin	TTL Signal	Signal Level
1 2 3	+ Keyboard Clock + Keyboard Data - Keyboard Reset (Not used by keyboard	+ 5 Vdc + 5 Vdc
	Power Supply Voltages	Voltage
4 5	Ground + 5 Volts	0 + 5 Vdc

5-Pin DIN Connector

Keyboard Interface Connector Specifications



Power Supply and Connectors

Mr. Kaypro

By Charles B. Stafford

Regular Feature

Kaypro Support

PC/XT Variants

Mr & Ms (dos) KayPro

Wherein Mr KayPro discourses upon MS (dos) KayPro, AND, Wherein, we discover that Non-Linear Systems actually did recognize that CP/M was not the only operating system in existence, that MS-DOS was not just a fad and would be a powerful force and, and that EVOLUTION is the key to survival, at least temporarily.

Early in 1983, it became apparent that Big Blue (IBM) had more influence on the micro-computer market than anyone had hereto fore realized. They were the largest supplier of mainframes to business, closely followed by Digital Equipment Co (DEC), and it was predictable that the "Information Systems" department of any large company would turn to a vendor they knew and trusted, when faced with the increasing clamor for desktop machines. Thus, some companies issued DEC VT-180s, CP/M machines, but most acquired and issued what became the ubiquitous IBM PC and subsequently the PC-XT, both running PC-DOS, an OEM version of MS-DOS. Only the small companies subject to stringent budget restraints opted for KayPros, Morrows, TRS-80s, and the like. Ultimately, they too would "upgrade" to PC-XTs or clones, as support for the others became less and less available and "compatibility" became more and more desirable. At this point the reason for the rise of PC/MS-DOS was largely economic, not because the operating system was technically superior. (Almost everyone has heard the story of how Gary Kildahl went flying instead of waiting for a telephone call from IBM, thus triggering the choice of MS-DOS for the operating system for the "new" PC instead of CP/M86. I'm told that the truth is that Gary was traveling the previous day and weather prevented him from returning that day, forcing him to return on the day of the fateful telephone call, and was in fact airborne when it came, otherwise we might not be having this discussion, but still all be using CP/M of one sort or another.)

Andrew Kay aka Non-Linear Systems recognized what was happening, and that the development of CP/M based software would slow down, if not stop altogether. His dilemma was that the manufacturing line was cranking out machines very efficiently, R&D already had a new model ready that was capable of graphics, and what do we do about the inventory, and how do we react if the market changes the way we think it might. The solution was to incorporate a third party product that would give the present model both CP/M and MS-DOS capability, and start development of an MS-DOS machine that would "maintain the KayPro character" (use as much of the existing engineering and inventory as possible). He achieved both objectives with the K-4+88 and the K-16.

The K-4s+88 were produced in both '83 and '84 models. The "+88" designation referred to an 8088 CPU on a co-processor board. The '83 models still used the 81-240 mother-board but the monitor ROM was labeled 81-292 SWP. SWP was the manufacturer of the co-processor board which was interfaced by using a daughter board between the Z-80 CPU and the mother-board. When the co-processor was in use, the Kaypro became an intelligent terminal and I/O channel. The co-processor board ram could also be used as a ram-drive for the Z-80 when running in CP/M. Needless to say, due to the limitations of the Kaypro video section, the machine had no graphics capability.

That little problem was solved with the issue of the K-4/84+88. This little jewel was almost identical to its predecessor, with the exceptions of an 81-184 mother-board and the marginally faster video section, and could handle limited graphics.

A more elegant solution appeared as the K-16. Loosely based on the K-10 and the IBM PC, it inherited the K-10 case, video section, power supply, floppy drive and hard-drive. Gluing all this together was a passive back-plane sporting 256k of ram, and three "cards"; a processor card with 8088, BIOS ROM and glue logic, a multi I/O card with serial and parallel ports, a floppy controller and another 256k of ram, and a monochrome/graphics video card. There was also one vacant slot for "whatever might please you." The back-plane was mounted upside down over the video section, with the cards parallel to the back, behind the video section. Major disassembly was required if you wanted to use that vacant slot, or replace a card. The K-16 worked very well and very efficiently, required a minimum of storage space when not in use, and was relatively easily transportable. Significantly, for the first time a KayPro was sold without bundled software, only the operating system and GWBasic. This continued until the issuance of the K-2000.

In response to requests for better transportability, and the relative fragility of the hard drives of the time, a second variant was produced, with two 360k floppy drives. The K-16 II was virtually identical to its predecessor with the exception of the substitution of the second

floppy drive for the hard drive, but it used a different monitor (boot) ROM which, to gain speed while booting, did not contain the code that would recognize and boot from a hard drive. Thus when users later added "hard-cards" they still had to boot from a floppy, although once booted the beast would access the hard drive properly.

About this time, the average computer user started the transition from being one of the "cognoscenti" or "literati" to being a "user", worse yet a "corporate user" with relatively unlimited resources, AND, color monitors started to appear. The PC-AT was rumored to be "in design" and the first "laptops" became available. The Bondwell and Epson were CP/M, using a 3.5" diskette and a microcassette respectively for storage.

Once more, Andrew Kay leapt into the forefront. The KayPro 286i was actually released before the IBM PC-AT, and was a truly amazing not to say impressive piece of hardware. The CPU box was huge, used an 80286 processor, and a type 2 hard drive (whatever that was) that was twice the capacity of anything running in an XT or a CP/M machine. What's more, it ran at the un-heard-of speed of 6 MHz, half again as fast as a PC-XT. A color monitor was optional.

At about the same time KayPro announced the K-2000, an MS-DOS laptop based on the NEC V-20 processor, which was not only 8088 compatible, but which would process CP/M instructions as well. The K-2000 was and remains unique. It was housed in an all aluminum case secured by tiny metric screws, had 784k of ram on board, had an 80 character by 16 line LCD display, a removable keyboard with cable for true "laptop" operation, about 2 hrs of battery, one 3.5" diskette, an optional modem, parallel and serial ports, and a header connector for an optional "docking base." The "docking base" contained a 5.25" floppy drive serial and parallel ports and had room for a hard drive, and a display card for a "real" monitor. When the K-2000 was on the "docking base" it could boot from the "docking base's" hard drive.

An "after-market"company advertised about a year later, coincident with the commercial appearance of back lit LCD panels, that it would, for a fee, "upgrade your K-2000 to a back lit screen. The only drawback was that battery life was cut in half.

A subsequent model of the K-2000 included an 80 character by 40 line back lit screen, but when the company declared Chapter 13 Bankruptcy to forestall creditors and liquidated it's remaining stock, only 80x40 non-back lit screens were available. The K-2000 and it's docking base were probably the most technologically advanced machines produced by KayPro.

The era just prior to the Chap 13 declaration was when the proliferation of "x" models happened and probably was an attempt to convert stocks of cases, motherboards and drives into cash. This was when the K-2x variants, K-4x variants, NEW K-2, and CP/M K-1 came into existence sporting combinations of motherboards, ROMs, drives, and operating systems. This is also when a real innovation occurred called the K-1 Student.

The K-1 Student is a three piece MS-DOS machine based on the NEC V-20 CPU housed in a case about 2 inches high, 10 inches wide and 10 inches deep. It came with an 84 key keyboard, a 12 inch monitor, and 2 720k 3.5 inch floppy drives. It was equipped with 2 serial ports and a parallel port, and a "wall wart" power supply. It was an elegant little solution for the student, very well made, could be configured for "zero desk footprint" by putting the monitor on a shelf, and using the specially configured feet to hang the CPU on "bulldog" hooks on the wall, leaving your lap for the keyboard and, wonder of wonders, it came with sufficient bundled software to keep a student well prepared if not amused. Unfortunately, it came about two years too late.

The KayPro Company did declare Chapter 13, and did successfully reorganize, but the really innovative period was over. As KayPro pulled itself out of the hole, it assembled several PC clones, XT, AT, and finally 80386 based. I have heard of, but not seen, models designated as KPV-20 and KPV-30. The names lead me to believe that they were based on the NEC V-20, and V-30 CPUs, plug-in replacements for the 8088 and 80286, respectively.

KayPro was not alone in succumbing to the "too little, too late" syndrome. A similar fate was suffered by Morrow, Otrona, Attache, Seequa, Osbourne, Northstar, and others.

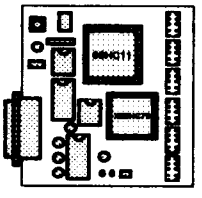
A FURTHER NOTE

I have not had the opportunity to explore any of the +88 models, nor do I have any of their operating systems. I would certainly appreciate copies of CP/M 2.2F, G, & H with the SWP enhancements.

Similarly I have not come across any documentation on the late MS-DOS KayPros and would appreciate copies of any printed matter anyone might have.

NEXT UP: External video for the '84 KayPros, heat up your soldering irons folks !!

$79.95 **68HC11**
Single Board Computer
SBC-8K
8K EEPROM for More Program Space!
SER-8C, Optional 8K Serial EEPROM $10



- Small Size, 3.3"x3.6"
- Low Power, <60 ma
- 8192 Bytes EEPROM
- 256 Bytes RAM
- DB-9 RS-232
- 24-TTL I/O Bits
- 8-A/D Inputs
- Power Reset Circuit
- 8 Mhz Clock
- Log Data with SER-8C

A Complete 68HC11 Development System.
New "CodeLoad+ 2.0" and Sample Programs.
No EPROMs or EPROM Programmers!
500 Pages of Manuals, 3.5" Utility Disk.

LDG Electronics
1445 Parran Road
St. Leonard, MD 20685
Voice / Fax
410-586-2177

Real Computing

By Rick Rodman

32-Bit Systems

All Readers

Linux for the 386/486

Toolkit for Linux

I've been using Yggdrasil Linux 1.1 around here, as I mentioned in previous columns. Linux is growing like wildfire in the Unix community, pretty much eclipsing Minix, Xinu and the other hobbyist operating systems. It is big - requiring a 386 processor and at least 4 megabytes of memory. Maybe now Minix can return to its roots as a small, fairly portable operating system for smaller systems.

Recently I acquired Walnut Creek's Toolkit for Linux, a 2-CD set which includes Slackware 2.0, which is a release of Linux 1.09. There are many differences between Slackware and Yggdrasil; the primary one is that Yggdrasil is *much* easier to install. Slackware is more for the technical Linux user.

Toolkit for Linux comes from Walnut Creek with 2 CD's and a slip of paper. The files on the disk explain how to install from floppies or from the Internet, but not from the CD. The steps needed are shown in Figure 1. The package offered by Just Computers is a newer release and includes a 600 page book, which may clear up any difficulties.

Slackware's installation is not as polished as Yggdrasil's. For experienced Linux users, it offers more flexibility; that flexibility can be a pitfall for novices. Where Yggdrasil only offers 4 configurations to choose from, Slackware puts almost every decision in front of you. Further, if you need to generate diskettes, for example, to read the CD on one machine and install on a second, you can do that very easily.

Slackware does nothing to help you set up X; Yggdrasil does only a little, which is a big difference. I couldn't get X working with Slackware, and I could with Yggdrasil. Still, both packages could use a *lot* of improvement in this area. *Suggestion to manufacturers:* Walk out of your office and find a novice user, and ask him or her to try to install your package. After you can pass that test, *then* you can say your package is comparable to commercial software.

Yggdrasil says its 'standard' installation will fit in 300MB; actually, it needs much more - it ran out of space with 440MB. This is a big jump from their lower setting, 'custom', which is 35MB.

While Yggdrasil is uncompressed on the CD so it can be run in "CD-dependent mode", nearly all of the files on the Toolkit for Linux are compressed. This means that you get much more than twice as much software with the Toolkit for Linux.

I prefer the ease of installation of Yggdrasil, but the tremendous amount of software included with Toolkit for Linix is too hard to resist. Just in programming languages, there's C, C++, Pascal, Fortran, Common Lisp, and Smalltalk. There's vi and emacs editors. There's a spreadsheet and several modem packages, and Internet mail and news utilities. And lots and lots of games.

The Toolkit for Linux includes archives from TSX-11 and "Sunsite". Within these archives are experimental versions of much new software for Linux, including the versions for the 680x0 (where x >= 2). There's lots of games, both character-mode and X Window, including a great Space Invaders clone (what system would be complete without that?). In fact, there's more software on it than you can shake a stick at. And I can shake a stick at a lot of software!

InfoMagic offers three more Linux CD packages, none of which I have seen, but all fairly inexpensive. No one said you can only have one. You can freely intermix the packages; Linux is still Linux; the only differences are the pieces included and the installation program. Consider this: Buying *both* Yggdrasil and Toolkit for Linux at list price is still less than one copy of Windows NT or OS/2 Warp. NT doesn't include an NFS server or client; Warp doesn't include any networking at all - and neither of them come with *all source code!*

Incidentally, it's still unclear exactly what hardware *is* supported by Linux. I've noticed that Future Domain SCSI boards are supported, some Trantor boards (often OEM'd by NEC and others) are, and, apparently, the Western Digital WD-7000 board. I have some of these and may try them out. A variety of non-SCSI CD-ROMs are also supported. I'm not set up to test the networking yet. A huge number of network cards are supported, including the venerable 3C501.

One other thing: It's almost always cheaper to buy CD software from a dealer than from the manufacturer. The dealers give you a discount; manufacturers charge full list price.

What's it like to use Linux?

Linux can be considered a superset of Unix, really. That is, pretty much anything that works under any version of Unix will work under Linux, and most

of the limitations of early versions of Unix, and Minix, are all gone.

Still, people who are familiar with MS-DOS may be confused, so I'll give some basic facts. First, there are no drives, and there's only one current directory at a time. Under DOS, you had a current directory under each drive; so, at first blush, this might seem to make copying files from place to place rather cumbersome. Also, since there's only one hierarchy, it gets considerably deeper than under DOS - I usually recommend going no deeper than 3 levels under DOS, while under Unix, it's difficult to avoid going six or more levels deep. There are a number of ways to avoid the horrific typing that would be otherwise necessary. The best one is that you can represent your home directory with the tilde ("~"). So, to copy an example directory like /usr/src/X11/mit/demos/xfish to a subdirectory in your home directory, you can cd to the directory and use "mkdir ~/xfish; cp * ~/xfish".

From this you can also see another difference between DOS and Linux: Wildcards work. For example, to unzip all the files in a directory, just enter "gzip -d *". (As I mentioned earlier, gzip, aka Gnu Zip, is replacing compress in the Unix world.)

Of course, you can have aliases for your popular commands. Aliases came from Unix originally, were implemented in Z-system, but for some reason never made it into DOS, OS/2 or NT. Presumably Microsoft didn't think that Joe Six-Pack, Typical PC User, was smart enough to use them. There's no PATH command (but you could make an alias). To see your path, use 'echo $PATH'.

As you know, Linux is multitasking. A very convenient way of making use of that is the virtual console feature. By pressing Alt-F1 to Alt-F4 (to Alt-F8 on Slackware), you can switch between any of four or eight virtual consoles - each with a different user logged in, for example. It's much easier to use than the background job stuff Unix folks are used to using.

The 'dosread' and 'doswrite' programs from Minix are no longer needed. Why? Because Linux supports FAT and HPFS natively. To access a DOS floppy, you just say 'mount -t msdos /dev/fd0 /mnt' and then access the files on the floppy in /mnt. Remember to 'umount /dev/fd0' when you're finished.

X Window is a lot different from Microsoft Windows. As I've mentioned before, it's much more complicated to get set up, and only serial mice seem to work. Then there's the matter of *window managers*. With Yggdrasil, the default is one called "fvwm", which seems to be rather Motif-like; a couple of others are included. The Toolkit for Linux includes several more. The Window Manager takes care of the "frame" of your windows - the outer parts, including the caption bar, system menu, and so on - and the interrelationships of the running windows. The programs themselves only deal with the "insides" of the windows. Thus, changing window managers can have an effect similar to moving between Windows 2.1 and Windows 3.1. (While you're in X, you can't switch virtual consoles. You can, however, start multiple 'xterm' windows, each with a system prompt.)

What's it like to program for X?

The easiest way to learn a new programming environment is to look at sample programs, taking them apart. For Microsoft Windows, you start with a program called GENERIC, which, in the original SDK, was heavily commented and included little extraneous material.

Unfortunately, the art of writing example programs is little practiced these days. Most sample programs you see are both poorly commented or uncommented, and include a lot of extraneous logic. For example, a sample program might perform computations of the number of Friday the 13ths in a given year, for example, with pieces of the logic scattered around the program, so it's impossible to distinguish which lines are essential.

The sample program which *ought* to be provided would have some fixed text such as a prompt, an edit window into which you could type something, and a couple of buttons. To meet this need, Listing 1 is the program 'Goodbye' from Asente and Swick, adapted to run under Linux. Listing 2 is the Makefile for goodbye.

Windows programmers will notice that, while everything they know in Windows has analogs in X, the differences are non-trivial. By the way, this is an X Toolkit program. In raw Xlib calls, it would be many times longer. Certainly it's already long enough, considering how little it does.

Until next time

Back in the early days of hobby computing, when 16K was a lot of RAM, we used to say "You're only limited by your imagination!" The Big Boys poohpoohed our little machines, and still do, but we knew better, and still do. The hobby computing spirit lives on in the pages of *TCJ*. Accept the challenge!

For more information

Real Computing BBS or Fax: +1 703 330 9049
E-mail: *Changing! To Be Determined*
Mail: 8329 Ivy Glen Court, Manassas VA 22110-4631

X Window System Toolkit, by Paul J. Asente and Ralph R. Swick, Digital Press (DEC), 1990

Walnut Creek CD-ROM (Manufacturer - Toolkit for Linux). 1547 Palos Verdes Mall Suite 260, Walnut Creek CA 94596, +1 510 674 0783, 1 800 786 9907

Yggdrasil Computing (Manufacturer - Yggdrasil Plug & Play Linux).4880 Stevens Creek Blvd. Suite 205, San Jose CA 95129, +1 408 261 6630 or 1 800 261 6630

InfoMagic Inc. (Manufacturer - three Linux packages). P.O. Box 30370, Flagstaff AZ 86003, +1 602 526 9565 or 1 800 800 6613

FIGURE 1: INSTALLING SLACKWARE FROM CD

```
Under DOS:
        Format 3 floppies
        E:                      (log to CD drive)
        CD \SLAKWA0\BOOTDSK1
        COPY SCSINET.GZ C:\TEMP
        CD ..\ROOTDSK1
        COPY COLOR144.GZ C:\TEMP
        CD ..
        COPY *.EXE C:\TEMP
        C:
        CD \TEMP
        GZIP -D SCSINET.GZ
        GZIP -D COLOR144.GZ
        RAWRITE
          SCSINET
          A:        (label this floppy BOOTDSK1)
        RAWRITE
          COLOR144
          A:        (label this floppy ROOTDSK1)
Shut down and boot the BOOTDSK1
        Press enter, asks for ROOTDISK
        put in ROOTDSK1, press enter
        Login as root
        fdisk /dev/sda
          n p 1 1 441 w
        mount -t iso9660 /dev/sr0 /mnt/cdrom
(Note: Later, you will have to use /mnt, as the cdrom subdir isn't created on the hard disk)
        setup
          T   (set target)
          S   (set source - use 4, mounted directory)
                /mnt/cdrom/slackware/slakware
                (Note: Later on, you'll have to omit the cdrom part)
          Proceed to select the packages you want
At this point you can run character-mode.  I never got X to work with Slackware, but there is
a program provided, briefly mentioned during setup, called ConfigXF86.  The directory
shown during setup is incorrect.  You invoke it like this:
        cd /usr/X11/lib/ConfigXF86
        ConfigXF86
If you make any wrong choice, and there is little guidance in making a correct one, you will
have to reboot and start over.
```

LISTING 1: GOODBYE.C

```
/* goodbye.c - from page 28 of Asente & Swick */
        /* Standard include files for X Toolkit applications */
#include <X11/Intrinsic.h>
#include <X11/StringDefs.h>
        /* Include files for classes of widgets we use */
#include <X11/Xaw/Box.h>
#include <X11/Xaw/Label.h>
        /* The Linux installation doesn't include 'Pushbutton'.  However,
                there is a 'Command' widget which appears similar. */
#include <X11/Xaw/Command.h>
        /* Callback function for the button widget.  We can call it anything. */
void Goodbye( widget, clientData, callData )    /* callback procedure */
Widget widget;                  /* analagous to HWND in Windows */
XtPointer clientData, callData;
{
        exit( 0 );      /* This button causes the program to exit */
}
        /* The following are arrays of arguments to the widgets.  The constants
        like 'XtNx' are macros which evaluate to strings.  That way, the compiler
        can verify your spelling; if you use the actual strings, they won't be checked
        until runtime. */

Arg labelArgs[] = {  /* The arguments to the label widget */
        {       XtNx,                   10      },
        {       XtNy,                   10      },
        {       "borderWidth",          0       },
          /* I didn't want a border on the label, so I looked in Label.h to find this string. */
        {       XtNlabel,   "Goodbye, world"    }};

Arg buttonArgs[] = {            /* The arguments to the button widget */
        {       XtNx,                   10      },
        {       XtNy,                   40      },
        {       XtNlabel,   "Click and die"             }};

/* ---- Main program ---- */

int main( argc, argv ) unsigned int argc; char **argv; {
        Widget          toplevel, label, button, box;
        XtAppContext    app;

        /* Call XtAppInitialize to initialize the X Toolkit library */

toplevel = XtAppInitialize(
                &app,                   /* -> Application context */
                "Goodbye",              /* Application Class */
                ( XrmOptionDescList ) 0,        /* command params */
                0,                      /* number of command params */
                &argc,                  /* -> count of args */
                argv,                   /* args on command line */
                ( String * ) 0,         /* specifying attributes */
                ( ArgList ) 0,          /* override values */
                0 );                    /* count of override values */
/* Should check return value in toplevel, I suppose. */

/* Create a 'box' managed widget - a child window, in Windows parlance. The widget name
is analogous to the window ID in Windows. */

box = XtCreateManagedWidget(
        "box",                  /* name of the widget */
        boxWidgetClass,         /* class of widget to create */
        toplevel,               /* parent widget */
        ( Arg * ) 0,            /* attribute values for widget */
        0 );                    /* number of attributes in list */
/* Again, we should check the return value */

/* Create a 'label' managed widget. */

label = XtCreateManagedWidget(
        "label",                /* name of widget - Not the text! */
        labelWidgetClass,       /* class of widget to create */
        box,                    /* parent widget is the box! */
        labelArgs,              /* array of arguments to label */
        XtNumber( labelArgs ));  /* number of args - we could just have said 3 */
/* Again, we should check the return value */

/* Create a button. */

button = XtCreateManagedWidget(
        "button",                       /* name of widget, not the text. */
        commandWidgetClass,             /* class of widget to create */
        box,                            /* parent widget is the box again */
        buttonArgs,                     /* array of arguments to button */
        XtNumber( buttonArgs ));  /* number of args - again, we could just have said 3 */
/* Again, we should check the return value */

/* Add a callback to the button, which is called when events occur */

XtAddCallback( button,    /* widget to add callback to */
        XtNcallback,            /* 'name of the callback' */
        Goodbye,                /* function procedure address */
        ( XtPointer ) 0 );      /* pointer to be passed as clientData */

/* Make all the widgets appear */

XtRealizeWidget( toplevel );    /* pass top-level widget */
XtAppMainLoop( app );           /* call dispatch function; in Windows,
                                        you normally code this in WinMain */
}
```

LISTING 2: MAKEFILE FOR GOODBYE

```
# link goodbye.o into goodbye.  I don't know which libs are really needed.

goodbye: goodbye.o
        rm -f goodbye
        gcc -o goodbye goodbye.o -O6 -m486 -DNO_ASM -fwritable-strings \
                -lXmu -lXt -lXaw -lXext -lX11 -L/usr/X386/lib -lm

# compile goodbye.c into goodbye.o.  I have no idea what all the -D lines are for.

goodbye.o: goodbye.c
        gcc  -O6 -m486 -DNO_ASM -fwritable-strings -I/usr/X11/include \
          -D_POSIX_SOURCE -D_BSD_SOURCE -D_GNU_SOURCE -Dlinux \
          -DFUNCPROTO=11 -DNARROWPROTO \
          -c goodbye.c -o goodbye.o
```


LINUX $61.95

Slackware Pro 2.2

Includes 4 CD-ROMs and a 640 page manual

A ready-to-run multitasking Unix clone for 386 and higher PC compatibles. TCP/IP, C, C++, X Window, complete Source Code, and much more!

JUST COMPUTERS!

(800) 800-1648

Fax (707) 586-5606 Int'l (707) 586-5600

P.O. Box 751414, Petaluma, CA 94975-1414

E-mail: sales@justcomp.com

Visa/MC/Int'l Orders Gladly Accepted

For catalog, send e-mail to: info@justcomp.com

Include "help" on a single line in the message.

Regular Feature

Intermediate

Trenton & Letters

Dr. S-100

By Herb R. Johnson

"Dr. S-100's Trenton ComputerFest column" by Herb Johnson (c) Apr 1995
Internet: **hjohnson@pluto.njcc.com**

Introduction

My new day job has eaten away at my free time - I mean my valuable S-100 development and testing time - so I'm a bit pressed this issue. But I've also continued to support other developments, and have the usual correspondence to report. There's been movements on a number of fronts, including a "front" I initiated two years ago! Also, as I write the Trenton Computerfest has just ended. Unfortunately, in all this I have not had a chance to turn on the Compupro 8/16 system. But I really really promise some writing on it next issue. I hope my readers will understand that, like themselves, we persue these things for personal interest, and with the desire to help others with similar interests: so occasionally our life interfers with our fun!

Trenton ComputerFest and GIDE

I can only report on small parts of this computer show, which celebrated its **20th year** of operation here in New Jersey, just 15 miles from my home in Ewing NJ (near Princeton). While there is a whole day of speakers on CP/M, I spend the **two days of flea market** out in the parking lot, roaming for yet another S-100 system. I did attend the **Zed Fest** for Z-System fans. This year, my interests and the Z-Fest coincided with the demonstration of a prototype **GIDE** interface from Germany. After the Saturday program, I joined the Z-fest at the traditional pizza dinner, and then we retired to the hotel to test the interface. Time and a lack of a good IDE harddrive limited the testing to a "go-no go" test. However, the prototype DID plug into the Z-80 socket on a Kaypro, and it DID receive some responses from the IDE drive.

I also received some information from Tilmann Reh as to costs for the prototypes. In addition, I recently began discussions with a local NJ computer and robotics company who imports systems from England. Their British counterpart has initially agreed to import the GIDE for at least one shipment, to "test the waters" for shipping more. For those of you who have not purchased equipment from Europe lately, you should know that there are "import duties" to pay on any equipment or electronics imported into the US from Europe, no matter how small or cheap. Fortunately the duties seem to be only a few percent. However, to avoid the cost and trouble of an import license, I must work with a licensed importer. In addition, there is the problem (and cost) of currency exchange, not to mention the transfer of money halfway across the world. This will also be handled by the agent.

Details are yet to be worked out and agreed to, but those people who showed interest in the GIDE over the last few months will be the first to be offered the device on a **prototype basis**, with minimal software but sufficient information to began development of software to support various systems. I'm also pleased to announce that Tilmann has shown interest in a variation for the **8085** processor at some point: this opens the door for many systems including the Heath/Zenith **Z-100** and **Compupro 8/16** systems.

Progress on any of these projects depends on the support of an active community of "classic computer" users. Access to that community depends, in turn, on publications like this one. Please keep that in mind when you consider renewing your subscription, or writing a letter, or even considering providing an article!

S-100 against IBM-PC's: no more

Bill Kibler asked me to "shine some light on XT's" in my article, as this issue will be devoted in part to the subject. Indeed, at the Trenton Computerfest I saw XT's stacked up like cordwood (and AT's as well!), most of them initially discarded for minor problems or simply for the (cyber)crime of being too slow. At the end of the day, many were bought and many were sold, and some simply discarded. While chasing the last few tailgate vendors for their "I don't want to take this home" bargains, I came across a pair of teenage boys; one of whom was carrying a **Z-100** on his shoulders. I asked about the computer. He said that someone had dumped it before they left, and there was another one near it. He only wanted the disk drive out of it, anyway. I offered to help him, and he offered to show me where the other system was.

We carried the loot to his mother's waiting car, and then I returned with him to pick up the other system, which had no hard drive in it but clearly used to. We chatted along the way, and he told me of the other computers and displays he and his brother had picked up this weekend for free, or nearly so. A 386 system from the dumpster that actually worked fine; the monitor with a bad cable; and so on. After we returned to his mother's car, I went to get my car and he went to get another load, running just ahead of the

"front loader" that the college provided to scoop up the junk left by the other vendors.

I returned with my car and talked to the mother, as I removed the hard drive from the Z-100 for the young man. She thanked me for the soda I gave her from my cooler, and told me how her son convinced her to take him around the neighborhood, just before trash day, to pick up old TV sets. "We'd get these TV's that people threw out, and my sons would make repairs, and sell them for maybe $50 each. They'd make quite a few dollars that way." I acknowledged grabing the occasional TV myself, and agreed how it was a shame that people threw stuff out because it was too expensive to have it fixed. It took me awhile to remove the harddrive, because I was using my wife's car and it didn't have the tools I needed. But I borrowed a screwdriver from another vendor (who watched me very carefully until I returned his tool), and then I returned with the hard drive to my friend's mother. We exchanged names and a handshake, and mentioned a few people we happened to know in common in her neighborhood, and then I bid her goodbye.

So what does this have to do with S-100 stuff? Namely, that the XT crowd is now in the same boat we S-100 people are in. We get our stuff from the same sources, we both have no vendor support, we both are dealing with technology that has been discarded and passed over. On the other hand, we both are finding that there is a lot to be learned from these old machine, that they can be understood and repaired, and that the pleasures (and a few dollars) we made as young folks on S-100 computer tinkering is being carried on by today's young folks with IBM-XT's.

Letters and E-Mail

I seem to have developed a regular correspondence with **Emmanuel Roche** in France, but I need to keep up my end! Careful readers of my column will remember a previous letter of his, where he provided some info on **Digital Research's GSX** graphics standard. But, he needed a graphics card from **Illuminated Technologies** to test some of his work on "deconstructing" graphics drivers. Well, I neglected to inform him that my recently-acquired Compupro system has just that card in it! Roche complains "about my lack of courtesy...by not warning me in advance!...May I remind you that I have disassembled several drivers for the NEC uPD7220 GDC (Graphics Display Chip)? This time, I won't give them to you before you ask: I will wait first for a letter."

"Still, you must understand that, having made so much work on this chip, I have a very high interest in this card and its screen, and its doc you were given along. I have never asked you so far to buy a complete computer because the price of shipping would be pretty expensive. But, I have two S-100 systems which could use this card, if it is a good portable S-100 card."

Well, I grovel in the dust at my thoughtlessness, Roche. I will send you a letter before I send off this column (well, before it gets published anyway) with a copy of the Illuminated Tech manual. Hopefully, together we can test and publish some of your GSX work on my card and system!

"To show you that I have nothing against you", he continues, "I enclose another contribution: the disassembly of UNLOAD version 2.0, as you are probably a user of CP/M 2.2." In fact I am, and UNLOAD is a nice little utility, so thanks! UNLOAD does the opposite of LOAD; namely, it converts a .COM file to a .HEX file, into the Intel HEX format. It can be convenient as a way to pass files between systems via serial links, as the Intel format is all characters, and has a checksum for each block.

Roche has another request: he needs Internet access, having moved to another city in France. He says there are only 3 IAP's (Internet Access Providers?) in all of France (?). He'd like me to ask my Internet server owner to help him launch his own server! Well, I certainly cheer him on to such a task! My provider (New Jersey Computer Connection) started out with a Linux system and a couple of dial-up lines (and a link to an Internet provider). He's now about to go T1 and maybe ISDN, and supports many more dial-ups, using a commercial UNIX. But Linux has a lot going for it. I'll suggest to Roche he find some Linux fans and refer him to the appropriate CD-ROM's. If any of my readers can help, please do so.

My series on the Compupro and the **Jade Bus Probe** evoked a response from John Maxwell in Buffalo NY:

" You probably don't remember me, but I purchased a couple hundred 8-inch diskettes from you several years back for my Imsai 8080 system. I was living in Lockport, NY at the time and I believe that you were living in Colorado somewhere. I had spoken to you on the phone and we talked and you sent me a copy of the Morrow "Thinker Toys" 8-inch controller manual.

"Anyway, I was pleased to see your name in "*The Computer Journal*" when I had first subscribed and am still enjoying your column. I look forward to it every month.

"This particular issue, *TCJ* #72, you mention a diagnostic tool that you use for troubleshooting S-100 systems. A Jade Bus Probe, is what I need to diagnose my Dynabyte and my other S-100 systems. You mention the Imsai's front panel to get an idea of what's happening in it, but is a pain to fire up two systems just to use one as a bus probe. If it wouldn't be too much trouble, could you send a copy of the schematic for the Jade Bus Probe? I don't know the current copyright status of such a device but I'd like to build one for use in my troubleshooting techniques. I have a few used proto-boards that I can add buffers and LEDs to. I was planning to use an extra board mounted to the top of the proto-board to mount the LEDs for ease of reading and labelling. I suppose I could homebrew it from scratch but if you have plans that you could share, it would save me valuable time between school and work and family demands.

"Of course, I'm open to any other suggestions that you may have that I could try as well. I'm glad to see that you're

still active in the S-100 field and perhaps I could 'bug' you again in the future to answer a few questions that I may have. Thanks for your time and hope to hear from you soon."

As far as I know, Jade Computers is no more (anyone know otherwise?). The board is somewhat interesting, so I'll send you a copy (not for free of course!). It's actually a nice accounting of all the S-100 bus lines: maybe I'll have Bill Kibler do a "Centerfold" of it for the readership [sounds good to me! BDK]. Of course, any other reader interested can contact me too. It's got a LOT of TTL drivers and LED's on it, folks!

Versafloppy progress

I've recently begun correspondence with **Ramon Gandia** in Nome, Alaska on a number of S-100 and Z180 subjects. He purchased a Versafloppy II controller from me, partly for the reason of sharpening his skills in BIOS writing for the **GIDE** interface he hopes to purchase; and partly to see if other floppy and IDE technology can be brought to the S-100 bus. Some of my readers may remember my series two years ago on writing a BIOS for the SD Systems Versafloppy II for 8-inch drives. With diligent efforts, Ramon has begun to write a version for **3.5-inch** drives. As these issues will be revisited for IDE work, I thought it was doubly important to mention them here in detail:

"Ok, I have the controller on the air. There are a few rough edges, but my boot, bios and sysgen are working reliably. Here's what I got. It runs 3.5" disks formatted as 26 sectors of 128 bytes each, single sided and 77 tracks. Sounds familiar? It is a double-density emulation of the standard IBM3740 format [standard for 8-inch drives]. My idea is to come up with a bios where drives A&B are 3.5 inch and the C & D drives are 8" for data interchange. Since I don't have any 8" drives, I am leaving that detail for later. Maybe much later.

I did not want to clutter things up with sector deblocking algorithms. Later, when I go ahead and add an IDE drive, I'll probably use 128 byte sectors as well. Actually, the IDE device has 512 byte sectors. I could (depending on the IDE interface) only access half of that sector, or all of it. But to tie to the BIOS, my idea is to waste all but 128 bytes out of each sector. It is not as crazy as you think.... I just got a 108 MB drive brand new for $100; and prices are dropping rapidly on these hard disks. Any reasonable size is overkill for cp/m....

How do the 3.5's work out? What rough edges? Well, I still have this problem with loading/selecting and deselecting. Right now, when the disk drive light comes on it stays on unless another drive is selected. There are some quirky things cp/m does....I had to insert diagnostic code to find out.

When cp/m FIRST accesses a drive, it does so thru SELDSK. But once it has done that, there are no further access to that routine until you switch to another drive. If I load my heads and select the drive on every read, and then deselect after every read, there is a lot of click-clack and that is no good. So I put in a counter in the CONIN routine. After cp/m sits on CONIN for a few seconds (adjustable), the drive deselects. Fair enough. But how to turn it on? SELDSK won't do, and I am reluctant to insert code into the read and write routines. I am working on this to see how I can smooth things out. I want a minimum of clicketty clack.

[The answer to this question, which I told Ramon in detail, was that all decisions about accessing the floppy drive are "delayed" until the BIOS receives the actual READ or WRITE SECTOR call. Calls to set the drive, track, and sector merely load local BIOS data registers with the appropriate information. The Select Disk BIOS call also sets up the appropriate "drive table" for the drive, which has all the BIOS information about that drive's disk format. Then, at READ or WRITE time, the drive, track, and sector is accessed. This is where you really need a good book on BIOS writing, or at least the Digital Research CP/M manuals.]

Speedwise, the 3.5 boots almost instantly; a lot quicker than I thought possible for a floppy. Much faster than my 486 PC on any initial access. But of course, file transfers and the like are slower than with larger sector sizes and do not compare to my Fulcrum [IMSAI] controller which buffers entire tracks....

Remember I told you on the phone I needed to replace the Fulcrum controller because it would eat disk directories at least once a day? Well, after I put in the Cromemco ZPU [Z80] card (at 4 MHz), those problems have disappeared on the Fulcrum.... I'll try 2-megs on it one of these days and see what happens.

I sure like working with these 3.5" floppies. I am going to investigate double-sided operation soon, but I am not sure how to implement [the order of tracks and number of sectors]. And what about skew? I am thinking of trying different skews. Once I find the optimum, rather than change the SECTRAN tables in the bios, I could FORMAT the disk to optimum skew. That way the BIOS and its DPB table would stay the same, and would handle a disk formatted EITHER way. This is interesting and worth exploring.

My FORMAT program right now is a mess. But at least it shows signs of life.... it produces disks that seem to work, but not reliably. I am missing something. The worse part is I cannot look at the disk with a microscope to see what I've formatted?! You format a disk and it is really anyone's guess as to what was written because there is no real way to read it back. Oh, I can read the fill bytes (E5) and the ID (read-long), but the gaps, etc which are so critical, are not really available. At least, not that I can determine from reading the 1795 [floppy disk controller chip] spec sheet. [I use DDT and hand code a simple routine to read the entire track to memory, then I can Dump it and check the whole thing, or re-write it to another track or disk. Actually only takes a few lines of code. I will look for it and publish the line later. BDK.]

The SD Sales manual is not really very good at explaining how the controller works. It took me a lot of testing to find

things out. A godsend was my monitor PROM in the Altair which has an absolutely fantastic monitor in it. I can examine memory, change it, compare it, verify it, move it, etc. I can also input and output to any port and go to any memory address. Here's a jewel: I can receive from a serial port into memory starting at address 0100. So I can send a program (binary or ascii) from my PC to the Altair over the serial port. No handshaking; it will use all available memory but protect the CCP. Thus files up to 50K more or less can be transferred. At the end of the transfer, you hit the key and it displays in decimal and hex how many pages to save. So if you run this program under cp/m, you can G0 to warm start cp/m and then A>SAVE 27 MYPROG.TXT or whatever. Since this is a ROM based routine, you can write and assemble programs in another computer and send them to the Altair where the monitor can execute at 0100 or Move it to another memory location. This means the computer can be loaded with ANY software even without a working cp/m.

The BIOS for the Versafloppy is ROM-based. I think it will fit okay with the monitor in 2K. Am also working on a entry point for the monitor which will display the current machine state, sort of like DDT's where you get the C0Z0M0E1P0 A=35F4 etc display. This routine is already debugged but I haven't ROMMED it yet; awaiting to see how much room all of this takes....

My Altair fits me like an old shoe....

Did you read Bill Kiblers editorial about his feelings on running or continuing *TCJ*....? About a week before I got my issue I sent him a letter about the very thing, because somehow I had read between the lines that he was heading that way. Absolutely distressing and alarming.

The bad part, is what he needs help on is not ARTICLES, but the mundane aspects of publishing a magazine. He is right, without a larger advertising base, he is going to be overloaded with work. We should perhaps try to help out by getting him more advertisers. I have told several vendors about *TCJ*....but no luck so far. Do you know anyone at JDR Microdevices or JAMECO or DIGI-KEY? Maybe they would like to help out. It would be cheap for them, helpful to the hacker community, and I suspect of more use to the advertiser than an mega-buck ad in such a horrible rag like PC Magazine....

We would be doing ourselves a big service by trying to keep Bill's magazine alive, or it is going to go the route of *Microsystems Journal*, *S-100 Journal* (where IS Sol Libes these days?) and *Micro Cornucopia*. *TCJ* is the end of the line and its kind. You've had a long association with *TCJ*, so you are better poised to address these issues.

What do you think?

Its melting here again, but so far, the snowbirds have not shown up. When those little birds show up, it is a sure sign of spring. Any day now. Today was just beautiful, about 40 degrees, sunny and calm. And a few inches less of snow left....

Papertape and Teletypes

This could be another column in itself, but last winter I produced **paper tapes** of old Altair software, courtesy of a papertape reader/punch from **Dan Slayton** of **Florida**. When I ran short, **Frank Wilson** from **Tomales, CA** sent me a few rolls. He got the tape from **Microfax** in Colorado. Frank writes: " If you hear of any **Teletype** enthusiasts please let me know. The ones I have are about to push me out of my residence!"

Frank should talk to **Claude Kagan**, who I just ran into at a computer show near NJ. He also has an enormous collection of Teletype and even earlier coding equipment! Some of us remember Claude from his **Sam 76** language days, and apparently there is still some action with that! Claude donated an IMSAI to me to "pass it on" to a person who would recognize and support its historical value. I have someone in mind, and I'll report more on that another day.

References

Frank J Wilson, PO Box 55, Tomales CA 94971
Microfax, 5620 F Kendall Court, Arvada CO 80002 (303) 467-1207
Dan Slayton, 1500 NE 40 PL, Oakland Park FL 33334
Emmanuel Roche, 8 rue Herluison, 10000 TROYES FRANCE
John H Maxwell <maxwell@cadman.cit.buffalo.edu>
Ramon Gandia <76241.3707@compuserve.com>
Bill Kibler <TCJ@psyber.com>

Do you need
Micro Cornucopia **Disks?**
Boot Disks?
Disk Copying?

Lambda Software Publishing

can now supply reprints of *Micro Cornucopia* Magazine, Kaypro Disks, Boot disks, CP/M 2.2, ZCPR and CP/M programs.

Kaypro disks	$5.00
all 49 disks	$200.00
Catalog of disks	$5.00
Disk Copying	$10.00
MicroC reprints	$8.00
Z-Letter back issues	$3.00
CP/M 2.2	$25.00
Spellbinder v5.3H	$60.00

Contact
Lambda Software Publishing
149 West Hilliard Lane
Eugene, OR 97404-3057
(503) 688-3563

Regular Feature

Editorial Comment

Building 8048 Systems

8048 Emulator

By J. G. Owens

THE STORY OF AN 8048 ETC. EMULATOR Part three.

THE SOURCE TEXT; "PORT" MACRO

The source for this program has gone through many changes. It started-out with my first 8048 assembler which I purchased from a vendor whose name may still appear in unflattering comments in the source. I will use that nameless cross-assembler as an excuse for any strange awkwardnesses in the code that may appear; at least one example is the upper-caseness of it, which is no longer required by the current MA48 assembler (see below).

It was subsequently converted to an assembler I hacked-up from Hendrix Small Mac assembler to handle the 8048; this conversion was tested in the emulator.

Finally most recently the amazing MA48 assembler and LA linker were used; the correctness of these translation tools was verified by creating a new binary file that matched the old one byte for byte.

There's a really huge PORT macro in XCOM.LB. I have no intention of figuring-out what I thought I was doing, but I don't think I'd do something like again. I mean if I really needed the functionality I should've just written a subroutine. Of course subroutines aren't easy in the 8048 either, so perhaps I knew better than I think. It seems to be used pretty thoroughly, and the general effect appears to be to relate the names of signals on the schematic to actions in the software, which is a Good Thing, so perhaps it all makes sense.

The source includes really astonishing comments (see the weird diagrams around "CO:"), and I hope they explain how the code works, because frankly I have no intention of going in there and figuring it out again so I can explain it here. Heaven knows, I wrote enough then....

BIT-BY-BIT SERIAL HANDLER

The monitor program controls the target according to commands it receives from an RS-232 link, presumably from the PC host program described later.

One of the things you can learn from X1.MAC is how to send and receive serial communications bit-by-bit without a UART chip. You can't do both at once; so the monitor doesn't. This works OK, since you want a send-command, get-response deal anyway.

Find "CO:" and "CI:" in X1.MAC to see how it's done on an 8048, which is an extremely stupid microprocessor. As you can tell from the comments, the link runs at 9600 baud, which isn't shoddy for this sort of lunacy.

THE HOST SOURCE

FILES: M48F.BAT Produces prototype include file M48.P using
GETFUNCS.EXE prototype program hopefully included somewhere around here.
M48.P Prototype file.
MON48.PRJ Turbo C 2.0 project.
M48LIB.H Include file.
MASM48.MAK Makes MON48 assembly files. (Borland Make 3.5?)
JGOVIDEO.ASM Assembly does a few harmless video things.
M48DAS.ASM.The 8048 disassembler is in assembler; I believe this is because I built it once somewhere in assembler for something else. At any rate, that's it; it's there.

M48ASM.C I never wrote this; you can't assemble 8048 code in MON48. So I lied. It's says "are you kidding?" when you try. Source files are: M48DMEM.C, M48DUMP.C, M48EMS.C, M48FUN.C, M48GO.C, M48HEX.C, M48LIST.C, M48LOW.C, M48MACH.C, M48MAIN.C, M48OPTS.C, M48SBR.C, M48SEND.C, M48SUBS.C, M48SYM.C, M48TRACE.C, M48VERS.C, M48XMEM.C, and MON48.EXE, the PC executable.

OPERATION

If you run MON48 without the emulator connected, it goes

```
mon48 version Nov 24 1993 [tc] (?:help; ^C stops)
nasty error: get48: missing input. Press <RETURN> to restart.
```

As indicated, pressing control-C seems to get out of it. Of course it does the same thing if the project for whatever silly reason doesn't work. But it really works; I just turned the poor thing on, and it still responds amazingly to commands. At any rate, when it works the response will be

```
mon48 version Nov 24 1993 [tc] (?:help; ^C stops) slave version ag.
@
```

With the "@" as its prompt. The "slave version ag" refers to the MONITOR program, specifically

```
"VERZ   EQU   256*'a'+'g'"
```

in XCOM.LB.

MON48 works like other monitors; when in doubt, I copied CP/M's DDT. Type a command, press Enter, and MON48 attempts to execute the command. If MON48 doesn't understand the command, it prints a question mark before the next @ prompt: "?@".

Like DDT, spaces aren't allowed between the command and the argument. Arguments and display are normally HEX numbers. When MON48 is outputting a dump or whatever, pressing any key will usually stop it.

It's easy to set-up MON48 for any particular start-up state you like using the MON48 batch file feature; see "COMMAND-LINE OPTIONS; BATCH FILES" below.

TO QUIT, Type Q, Return.

COMMANDS

Once it's working, if you type ?, Return, it prints

```
C:\MON48:[ov] mon48
mon48 version Nov 24 1993 [tc] (?:help; ^C stops) slave version ag.
@?
'<string> ~ sends, prints return. <;> 3G[start][,break] ~ Go; key breaks.
= RET. <*> = BEL = error.                    3K[a] ~ Kill break address in G.
D[a1][,a2] ~ dump.  L[a1][,a2] ~ list.3T[steps] ~ Trace.
Q ~ quit.     ^C ~ quit, don't stop.3O1y ~ Turn on op 1; "O1" or "O1n":
S<a> ~ substitute; . terminates.     3 off. "O": menu, u can type then.
X<a> ~ external (outl or movx); .    3E BUS | E BUS INPUT | E BUS OUTPUT ~
terminates.                                  3 specify bus emulation.
R0..R7F, R0'..R67' ~ int. data memory.3Y<code offset> ~ set dot offset.
.PC .A .PSW .P1 .P2 .P4 .P5 .P6 .P7 .T3[op1][op2] ~ Exec 1 or 2 ops in
.C .T0 .T1 .F0 .F1 .INT .RB .MB ~   3 target.
registers; . terminates.                     3+ ~ Display data sent to emulator.
H<f> ~ load hex. Z<f> ~ sym.; then 3** Restart.
Name/;Name instead of #; Z- clears.          3?[a1][,a2] ~ This menu, # arg, args:
you typed: ? count: 0 arg1: 0 arg2: 0
```

I will now describe each of these commands, *for the very first time anywhere!*

```
'<string> ~ sends, prints return. <;> = RET. <*> = BEL = error.
```

The apostrophe introduces a string which MON48 will send to the MONITOR program in the emulator, and then print the return string sent back. As noted, the semicolon can be used to simulate a RET, and the asterisk simulates the BEL character. The commands that the MONITOR understands are documented in the monitor source. This command is obviously included for purposes of debugging the monitor.

```
D[a1][,a2] ~ dump.  L[a1][,a2] ~ list.
```

Dump and List commands which work like CP/M's DDT; "a1" and "a2" refer to hex addresses. Here's a list screen:

```
@L100
100:    SEL     RB1
101:    MOV     R2,#14
103:    CALL    13E
105:    MOV     A,#FE
107:    OUTL    BUS,A
108:    MOV     R3,A
109:    MOV     R4,#01
10b:    MOV     R1,#20
10d:    MOV     R5,#08
@
```

Q ~ quit. ^C ~ quit, don't stop.

To leave the program *and* stop any ongoing emulation, Q. To leave the program and leave any running emulation going, use ^C (Control + C).

```
S<a> ~ substitute; . terminates.
```

Again, works like the DDT command. Here's what the screen looks like in a substitute command, followed by a D:

```
    mon48 version Nov 24 1993 [tc] (?:help; ^C stops) slave version ag.
    @s100
    100: EA ab
    101: FF cd
    102: 03 ef
    103: F9 .
    @d100
    100: AB CD EF F9 00 FF 00 FF    00 FF 00 FF 00 FF 00 FF  ................
    110: 01 FE 00 FF 00 FF 08 FF    01 FF 00 FF 00 FF 01 FF  ................
[... etc.]
    @
```

```
X<a> ~ external (outl or movx); . terminates.
```

You might have memory or a device connected to the 8048's data bus, so the X instruction lets you muck with that, working like the S instruction.

```
R0..R7F, R0'..R67' ~ int. data memory. .PC .A .PSW .P1 .P2 .P4 .P5 .P6 .P7 .T
.C .T0 .T1 .F0 .F1 .INT .RB .MB ~
registers; . terminates.
```

You can of course modify the 8048's internal registers. The 8048 has two banks of 8 registers, i.e. the assembly language lets you do things like "MOV A,R7". "SEL RB1" and "SEL RB0" switch between the two. Both sets are actually mapped into the tiny internal RAM space at different locations (you can only access the rest of 8048 RAM with indirect instructions using an index within a register).

So MON48 pretends that the register set runs up to R7F, allowing you to inspect each and every byte of the maximum 128 bytes possible. Again it works just like the substitution command. The usage "R0'" refers to the alternate set, and works really the same way, except that 0 is translated to 18H, which is where the alternate set begins.

Finally you can examine and modify the registers and various other bits and pieces of the 8048 condition; for instance, there is an instruction "JNT0" which will jump if the 8048 pin named T0 is *low* (of course there isn't any instruction for when it's high). So you can examine T0 like

```
@.t0
T0: 0 1
```

```
?T0: 0
```

but you can't modify it; see, MON48 prints a ? and restates the situation, just in case you forgot.

H<f> ~ load hex. Z<f> ~ sym.; then Name';Name instead of #; Z- clears.

Well the whole reason for the thing is so you can download your test file, and it should be in Intel hex format. *And* you can download a symbol file! The symbol file should be in a format like

```
begin                   000000DC
rsmtop                  00000040
nocom                   000000B4
tkey                            00000300
```

i.e., symbol name, hex value. MA48 produces these ("-sFILE.SYM"), or you can use the linker's output (also "-sFILE.SYM"), but then you'll only get the public names. MON48 apparently understands the comments LA embeds in the symbol file and ignores them.

Once you do this, you can use a symbol wherever you used a number, or at least that seems to be the way it works, i.e. "Dbegin" does indeed dump starting @ 00DC. The alternate usage with a leading semicolon is apparently supposed to cover names that would be ambiguous with hex numbers, i.e. "ABC" might be a legitimate label, so you go "L;ABC" and MON48'll understand.

G[start][,break] ~ Go; key breaks.

Like DDT; an optional start address and a single break address is permited. The emulator sets your program off, from the start address if specified, to run until the break if specified. When/ if the break is reached, the target program will stop executing at the instruction *after* the break. If for instance you set a break at 0DC ("G,DC") and at that location there is

```
"00DC: CALL 7"
```

then the breakpoint will occur at 007.

Pressing a key at any time will stop emulation; the MONITOR will stop the TARGET wherever it is, and then tell you where it stopped.

If you want to leave the program running and exit MON48, use ^C.

K[a] ~ Kill break address in G.

See the discussion above "DO BREAKPOINTS ALWAYS WORK CORRECTLY?". In such a situation, MON48 actually provides a remedy, sort-of. Here's some actual code; when the program runs starting at 0, it goes somewhere and then calls a subroutine at 7, which returns at location 17, as you can see.

```
@l17
017:    RET
018:    CLR     C
019:    CALL    030
01b:    CALL    030
01d:    CALL    031
01f:    CALL    030
021:    CALL    030
023:    CALL    030
025:    CALL    030
@g0,18                          [A]
0de
@kde                            [B]
@g0,18                          [C]
bad break @ 0de
140
@
```

So if you wish to trap execution that occurs at location 18, you might "g0,18" as shown at [A]. *But* the MONITOR hardware traps what it thinks is execution at location 18, when the RET instruction at 17 executes. The breakpoint shown — "DE" — is where the monitor catches-up with the program, i.e., at the *next* instruction, which is the location after the call that the RET is returning from. I.e., at 0DC there's this code:

```
@Ldc
0dc:    CALL    007
0de:    NOP
```

So, you then use the amazing "K" instruction as at [B] to tell MON48 to ignore breaks at 0DE. And so it does, as shown at [C]. This is not a real-time situation; quite a long time passes as the MONITOR informs MON48 that it got break, and MON48 tells the MONITOR to please ignore and just keep going. But in many debugging situations that won't be important, so it's something.

T[steps] ~ Trace.

It'll trace one instruction, or the (hex) number you specify. Unlike DDT, MON48's trace doesn't blather a lot; it just prints the PC value as it goes along.

```
@t10
0dc 007 008 00a 00b 00d 00f 010 011 00f 010 011 00f 010 011 00f

@
```

That shows the program, which started at zero (to set the PC for a trace if you don't want to use the current value, use the .PC command) jumping to 0DC, and then calling 007, and executing there.

O1y ~ Turn on op 1; "O1" or "O1n": off. "O": menu, u can type then.

This is an option menu. If type the "O" command without arguments, it'll show you things you can do:

```
@o
1Y:port 1 is input (restore to ff at G).
2Y:port 2 is input (restore to ff at G).
3Y:don't break at IBM keypress during execution (G); instead of using serial
        to talk to emulator, print incoming serial on IBM screen.
4Y:print option 3 serial data in hex.
(i.e., "1" or "1N": don't restore port 1 to ff).
(now: port 1 output; port 2 output; serial exec off)
```

You can then type-in one option. Or if you know what you want, you can use the form "o1y" for instance. As it notes, entering "o1" is equivalent to "o1n".

What it means:

1 and 2 control MON48's treatment of the 8048 P1 and P2 ports. The general idea is when you return to a program, MON48 tries to restore the registers as they were. If the port is an input port, this may not be a good idea at all, so you have this control. The control really should include some kind of mask, because *some* of the bits may be inputs etc.

Options 3 and 4 take care of the case where I wanted to debug a TARGET with serial communications. The idea is I'd go "o3y", "G", and then switch cables somewhere, so I could see the amazing stuff it was emitting. When you do that, MON48 says

```
bugger active: printing serial received until IBM keypress='x'.
be sure emulator's plugged-in before pressing 'x'....
```

which is reasonable advice. Obviously this option needs a little work depending on the particular target application — but then that's why you have the source!

This option information is *not* retained in the MONITOR, but is transient data maintained by MON48 only as long as it is running. Use a batch file to set these things up if necessary (see "COMMAND-LINE OPTIONS; BATCH FILES").

E BUS | E BUS INPUT | E BUS OUTPUT ~ specify bus emulation.

Really like the O1 and O2 options, but for the 8048's BUS port. In this case the bus normally *must* be used as one of these three, you can't divide-up the bits and use them differently, so this range of emulation should be OK.

This information is *not* retained in the MONITOR, but is transient data maintained by MON48 only as long as it is running. Use a batch file to set these things up if necessary (see "COMMAND-LINE OPTIONS; BATCH FILES").

Y<code offset> ~ set dot offset.

In the olden days, one would often have a listing for something at the *wrong address*. Even in the new days, I suppose. You can set a "dot offset" with the Y command, and then subsequently when any quantity (or symbol!) is prefixed with a period, it'll get the dot offset added to it. The dot offset, if non-zero, is printed before the @ prompt in brackets.

/[op1][op2] ~ Exec 1 or 2 ops in target.

Well the idea is use the MONITOR's instruction-force method for forcing your own instructions. Probably a bad idea, and was presumably included so I could test the mechanism.

+ ~ Display data sent to emulator.

Well it does that; tremendously interesting. It toggles; use it again to stop the effect.

?[a1][,a2] ~ This menu, # arg, args:

The MON48 help. As it notes, you can add arguments, and it tells you what it thinks they were (that's the bottom of the menu where it says "you typed: ? count: 0 arg1: 0 arg2: 0"), which was presumably helpful in debugging that mechanism.

MON48 COMMAND-LINE OPTIONS; BATCH FILES

If you go "mon48 ?" it prints this helpful information:

```
mon48 version Aug 26 1994 [tc]
-bfile  do batch commands I guess.
-s2  set send add-on delay in us.
-z500  set # of symbols.
```

Taking the worst first, "-s" adds a serial transmission delay; I think I added this around 11/93 when I resurrected the unit and my 386 overwhelmed the poor monitor. But the resulting default value — 2 — is ridiculous, since a 9600 character time is 1 millisecond or so. However, the microsecond timer is probably grossly inaccurate, so it all works out somehow.

"-z" allocates varying numbers of symbols. This is 8048 code, and you shouldn't ever have too many symbols.

Finally, and best of all, specifying a file with the "-b" will cause MON48 to attempt to execute the material in the file as if they were command lines. For instance, my fabulous MIDI pedal project has a file called PH like this:

```
;\pedal\sw\p:   set-up for mon48 for pedal software.          HEX LOAD.
o2y
e bus output
hpedal.hex
spedal.sym
```

and I can go "mon48 -bph" and it'll all happen! I have an alternate file without the hex load — which takes a little while — when I know it's already been downloaded. This is pretty hi-tech stuff here....

MA48 8048 MACRO ASSEMBLER

FILES: MA???.ZIP The assembler package.

The ???s are a three-letter date code like FHL or something. This assembler is keen; it does everything but slice bread. Documentation's in there.

SUMMARY AND APOLOGIA

Well, it works. As I wrote this documentation, I discovered several bugs — which is actually a procedure I highly recommend: writing the documentation *often* turns-up slews of bugs....

But I'm sure there are more. I mean, the whole thing is so strange and amazing, if you should ever get so far as to be able to use any of this, just remember when a bug bytes you — what a *venerable* bug it is!

— j.g. owen

Special Feature

68HC11,Support

Forth Based Tools

Small Tools

By Calvin McCarthy

Part 2 of Small Tools

A quick look at the New Micros Inc. F68HC11, the "Charm" for Low-buck controller developers.

The Motorola M68HC11 is an exciting microcontroller which embodies all the simplicity and power of the Motorola 6800 family and adds Input/Output capabilities to the package. New Micros Inc. added Forth in internal ROM and created the F68HC11, completing the charm for us low buck developers and hobbyists. This F68HC11 with a built in high level language makes a great foundation for a controller project.

Features:

F68HC11 microprocessor (v3.3 Integer Forth, v3.5 Floating Point)
Memory:
- RAM - 256 or 512 bytes
- ROM - 8k (Integer Forth) or 12k (Floating point)
- EEPROM - 512 bytes

Input/Output ports:
- Digital - Channel A: 3 bits input only, 4 bits output only, 1 bit input or output
- Channel B: 8 bit output only (or address and data bus expansion)
- Channel C: 8 bit configurable input or output (or address and data bus expansion)
- Channel D: 2 pins share input/output or asynchronous serial communications 4 pins share input/output or synchronous serial communications
- Analog - Channel E: 8 channels of 8 bit analog/digital conversion or digital input
-Timers - Internal 16 bit Timer with timer input capture and timer output capture functions and real time clock interrupt.

Why M68HC11?

The Motorola microcontrollers are very accessible. I find them simple to understand and easy to program. They have built in capabilities for programming the internal EEPROM so you do not need a complex development system. I was able to develop simple applications such as a slide projector advancer and a nicad battery charger with no extra support tools other than the New Micros Inc NMIS-0122 single board computer and the Pygmy Forth based tools on my PC.

Why F68HC11? It provides both a High level language and machine monitor. When Forth is added to the 68HC11 you get a high level language and a powerful monitor program all in one. You have all the benefits of a structured language with its decision capability (IF-ELSE-THEN), loops (DO-LOOP, BEGIN-WHILE-REPEAT, BEGIN-UNTIL), and constant and variable names. As well you have monitor like functions such as @ ("Fetch") to inspect a 16 bit memory location, C@ ("CFetch") to inspect an 8 bit location, ! ("Store") to put a 16 bit number in a double byte memory location, C! ("CStore") to put an 8 bit number in a byte memory location, FILL to fill a set of memory locations with a particular number, and DUMP to inspect a set of memory locations. All this is immediately accessible as soon as you connect your terminal and turn on the computer.

Self Programming

New Micros Inc. have added a few 68HC11 specific words to their Forth which, to my mind make the F68HC11 the charm. As soon as you turn on the power you have EEPROM programming capability. The word EEC! ("EECStore") stores an 8 bit number in either internal or external EEPROM. The word EEWORD takes a Forth word you have defined in RAM and places it in either internal or external EEPROM, adjusting all the threaded code pointers appropriately. You do not have to lose the words you have just defined when you shut down for the day and you do not need a special programmer.

There are times when you will want to use Forth code when normally you would expect to need machine code. For instance, the interrupt capabilities of the 68HC11 normally operate outside the Forth environment. It is assumed that interrupt code will appear in the machine as machine code. Now what if you are a confirmed Forth addict and you do not want to write your interrupt routines all in assembler? What if you want to try out your idea quickly before you code it in assembler to gain the added speed? There is a special word to help you here. The word ATO4 lets you hook a Forth word to assembler code. The interrupt code jumps normally to machine code, the machine code sets up the entry conditions for the word ATO4 then ATO4 manages the call to and return from the Forth word. See the AUTOSTART word described below for one use of ATO4.

Autostart

The built in autostart capabilities allow the system to initialize itself the way you want on startup, running a stand alone program or just reconfiguring the Forth for your convenience. On startup the machine looks at memory location $B600 (then

at every 1K boundary) for either $A55A or $A44A. If it finds either of these flags it looks at the 2 byte number immediately following for an address pointer. This will be the address of the code to automatically run. If the flags are not found the machine will find and start Forth.

Autostart can be used to overcome a number of limitations. The F68HC11 v3.3 comes with only 16 bytes of text input buffer (TIB). The TIB can be changed to reside in external RAM so it can be as big as you want (80 bytes?). Internal RAM is limited. The Dictionary Pointer (DP) can be changed to build the dictionary in external RAM if external RAM is available. The EEPROM dictionary pointers must be initialized if you want the words you have moved to EEPROM to be recognized the next time you turn on your machine. Also, you may want to set up other special parameters before you start.

How I use AUTOSTART. The following code is my autostart word called CONFIG. It was written as machine code using CODE-SUB but it would have been just as easy to use Forth. The code depends on the following conditions:

1. 8 K of Extended RAM beginning at $0100.
2. 8 K of EEPROM beginning at $8000.
3. The Dictionary Pointer is in location $001E.
4. The Text Input Buffer (TIB) pointer is in location $001C.
5. The length of the TIB is found in location $001E.
6. Pointer to the last word in EEPROM is retained in EEPROM location $9FFE.
7. Pointer to the next open EEPROM location is retained in EEPROM location $9FFC.
8. Pointer to extra initialization word is retained in EEPROM location $9FFA.
9. Location $9FFA is set to $FFFF if there is no extra initialization word.
10. The word CONFIG is located in the EEPROM

```
HEX
CODE-SUB CONFIG
96 C, 2C C,         ( LDA $002C  GET DP TO SEE IF MOVED )
26 C, 1E C,         ( BNE to RTS -  $2C IS #00 IF DP NOT MOVED )
FC C, 9FFE ,        ( LDD $9FFE LAST WORD IN EEPROM     )
DD C, 38 C,         ( STD $0038   SET IT                )
FC C, 9FFC ,        ( LDD $9FFC NEXT OPEN EEPROM MEMORY )
DD C, 30 C,         ( STD $0030   SET IT                )
CC C, 0100 ,        ( LDD #$0100  DESIRED LOCATION OF TIB    )
DD C, 1C C,         ( STD $001C   SET TIB LOCATION TO $0100  )
CC C, 0080 ,        ( LDD #$0080  TIB 128 CHAR LONG )
DD C, 1E C,         ( STD $001E   SET TIB LENGTH          )
CC C, 0200 ,        ( LDD #$0200   DESIRED LOCATION OF DP    )
DD C, 2C C,         ( STD $002C   SET DP LOCATION TO $0200   )
CC C, 0180 ,         ( LDD #$0180  DESIRED LOCATION OF PAD   )
DD C, 22 C,         ( STD $0022  SET PAD LOCATION TO $0180   )
FC C, 9FFA ,    ( LDD $9FFA   CFA FOR EXTENDED AUTOSTART )
1A83 , FFFF ,       ( CMPD #$FFFF  CHECK VALID CFA      )
27 C, 03 C,         ( BEQ RTS  IF NO EXTENDED AUTOSTART )
BD C, ATO4 ,        ( JSR  ATO4   )
39 C,               ( RTS         )
END-CODE
```

This code moves the Dictionary Pointer to $200, the TIB to $100, the PAD to $180, and sets the TIB length to 128 characters. As well it sets up the pointer to the last word in EEPROM by getting it from $9FFE and putting it in $0038 and the pointer to the next open EEPROM dictionary location by getting it from $9FFC and putting it in $0030. This sets up the EEPROM dictionary pointers so that all words in EEPROM will be available. Any new words will be put in the RAM dictionary starting at $200 until you move them to EEPROM.

The last action of CONFIG is to check memory location $9FFA to see if there are other parameters which can be configured. The number found in $9FFA is put into the D accumulator. If D now contains #$FFFF there is no more to do and the branch goes out to RTS. If $9FFA contains anything else the value found is considered the Code Field Address (CFA) of the high level Forth word which will provide the extra configuration action. This word is executed, then CONFIG exits.

NOTE:
1. You must make certain that you take the value stored in $0038 and save it to $9FFE and the value stored in $0030 and save it to $9FFC when you add words to the EEPROM dictionary.
2. You must be sure that you do not have any other words in RAM when you move a word to EEPROM and save these values.)

New Micros Inc.

As with any computer, the F68HC11 is not a "piece of cake". You will need the manuals to be able to make it work well. I was pleased with the detail in the manuals. The Forth words are described clearly both in summary and in detail. Various procedures like autostart, ROMing the code, Input/Output, and interrupts are clearly discussed. The memory map is provided clearly. I purchased the NMIS-0122 Single Board Computer and RS-232 interface and I was sent full schematics and documentation for these. Without the documentation I would have been lost. With it and a hacker's knowledge of microcomputers I was able to easily get control of the product.

New Micros Inc. offers the F68HC11 in a number of Single Board Computers as well as a wide variety of peripheral boards to complement the SBCs and some software tools. The SBC I received was up-to-date, very professionally executed and very competitively priced in comparison to other M68HC11 systems advertized with equivalent features. It would be good for you to get the product list and see what you could use. I do not think you could build either the SBC or the peripheral boards more economically than NMI can provide them. Certainly you could not improve on their quality.

Conclusion

I would encourage you to consider using the New Micros Inc. F68HC11 and their single board computers when you have an embedded control problem. With the embedded Forth and the software development tools described in the earlier part, you should be able to get a solution working in little time. And that is the name of the game.

New Micros Inc.
1601 Chalk Hill Road,
Dallas, Texas 75212, U.S.A.

Regular Feature

Contact Listing

SUPPORT GROUPS FOR THE CLASSICS

TCJ Staff Contacts

TCJ Editor: Bill D. Kibler, PO Box 535, Lincoln, CA 95648, (916)645-1670, GEnie: B.Kibler, CompuServe: 71563,2243, E-mail: B.Kibler@Genie.com, tcj@psyber.com.

Z-System Support: Jay Sage,1435 Centre St. Newton Centre, MA 02159-2469, (617)965-3552, BBS: (617)965-7259; E-mail: Sage@ll.mit.edu. Also sells Z-System software.

32Bit Support: Rick Rodman, BBS:(703)330-9049 (eves),.

Kaypro Support: Charles Stafford, 4000 Norris Ave., Sacramento, CA 95821, (916)483-0312 (eves). Also sells Kaypro upgrades, see ad inside back cover. CompuServe 73664,2470 (73664.2470@cis).

S-100 Support: Herb Johnson, CN 5256 #105, Princeton, NJ 08543, (609)771-1503. Also sells used S-100 boards and systems, see inside back cover. E-mail: hjohnson@pluto.njcc.com.

6800/6809 Support: Ronald Anderson, 3540 Sturbridge Ct., Ann Arbor, MI 48105.

Regular Contributors:
Dave Baldwin, Voice/FAX (916)722-3877, or DIBs BBS (916) 722-5799 (use "computer", "journal", pswd "subscriber" as log on), Internet dibald@netcom.com, CompuServe 70403,2444.

Brad Rodriguez,Box 77, McMaster Univ., 1280 Main St. West, Hamilton, ONT, L8S 1C0, Canada, Genie: BJ, E-mail: bj@genie.com.

Frank Sergeant, 809 W. San Antonio St., San Marcos, TX 78666, E-mail: fs07675@academia.swt.edu.

Tilmann Reh, Germany, E-mail: tilmann.reh@hrz.uni-siegen.d400.de. Has many programs for CP/M+ and is active with Z180/280 ECB bus/Modular/Embedded computers. Microcontrollers (8051).

Helmut Jungkunz, Munich, Germany, ZNODE #51, 8N1, 300-14.4, +49.89.961 45 75, or CompuServe 100024,1545.

Ron Mitchell, Apt 1107, 210 Gloucester St., Ottawa Ontario, Canada, K2P 2K4. GEnie as R.Mitchell31, or CompuServe 70323,2267.

USER GROUPS

Connecticut CP/M Users Group, contact Stephen Griswold, PO Box 74, Canton CT 06019-0074, BBS: (203)665-1100. Sponsors Z-fests.

Sacramento Microcomputer Users Group, PO Box 161513, Sacramento, CA 95816-1513, BBS: (916)372-3646. Publishes newsletter, $15.00 membership, meetings at SMUD 6201 S st., Sacramento CA.

CAPDUG: The Capital Area Public Domain Users Group, Newsletter $20, Al Siegel Associates, Inc., PO Box 34667, Betherda MD 20827. BBS (301) 292-7955.

NOVAOUG: The Northern Virginia Osborne Users Group, Newsletter $12, Robert L. Crities, 7512 Fairwood Lane, Falls Church, VA 22046. Info (703) 534-1186, BBS use CAPDUG's.

The Windsor Bulletin Board Users' Group: England, Contact Rodney Hannis, 34 Falmouth Road, Reading, RG2 8QR, or Mark Minting, 94 Undley Common, Lakenheath, Brandon, Suffolk, IP27 9BZ, Phone 0842-860469 (also sells NZCOM/Z3PLUS).

L.I.S.T.: Long Island Sinclair and Timex support group, contact Harvey Rait, 5 Peri Lane, Valley Stream, NY 11581.

ADAM-Link User's Group, Salt Lake City, Utah, BBS: (801)484-5114. Supporting Coleco ADAM machines, with Newsletter / BBS.

Adam International Media, Adam's House, Route 2, Box 2756, 1829-1 County Rd. 130, Pearland TX 77581-9503, (713)482-5040. Contact Terry R. Fowler for information.

AUGER, Emerald Coast ADAM Users Group, PO Box 4934, Fort Walton Beach FL 32549-4934, (904)244-1516. Contact Norman J. Deere, treasurer and editor for pricing and newsletter information.

MOAUG, Metro Orlando Adam Users Group, Contact James Poulin, 1146 Manatee Dr. Rockledge FL 32955, (407)631-0958.

Metro Toronto Adam Group, Box 165, 260 Adelaide St. E., Toronto, ONT M5A 1N0, Canada, (416)424-1352.

Omaha ADAM Users Club, Contact Norman R. Castro, 809 W. 33rd Ave. Bellevue NE 68005, (402)291-4405. Suppose to be oldest ADAM group.

Vancouver Island Senior ADAMphiles, ADVISA newsletter by David Cobley, 17885 Berwick Rd. Qualicum Beach, B.C., Canada V9K 1N7, (604)752-1984.

Northern Illiana ADAMS User's Group, 9389 Bay Colony Dr. #3E, Des Plaines IL 60016, (708)296-0675.

San Diego OS-9 Users Group, Contact Warren Hrach (619)221-8246, BBS: (619)224-4878.

ACCESS, PO Box 1354, Sacramento, CA 95812, Contact Bob Drews (916)423-1573. Meets first Thurdays at SMUD 59Th St. (ed. bldg.).

Forth Interest Group, PO Box 2154, Oakland CA 94621 510-89-FORTH. International support of the Forth language, local chapters.

The Pacific Northwest Heath Users Group, contact Jim Moore, PO Box 9223, Seattle, WA 98109-0223.

The SNO-KING Kaypro User Group, contact Donald Anderson, 13227 2nd Ave South, Burien, WA 98168-2637.

SeaFOG (Seattle FOG User's Group, Formerly Osborne Users Group) PO Box 12214, Seattle, WA 98102-0214.

OTHER PUBLICATIONS

The Z-Letter, supporting Z-System and CP/M users. David A.J. McGlone, Lambda Software Publishing, 149 West Hillard Lane, Eugene, OR 97404-3057, (503)688-3563. Bi-Monthly user oriented newsletter (20 pages+). Also sells CP/M Boot disks, software.

The Analytical Engine, by the Computer History Association of California, 1001 Elm Ct. El Cerrito, CA 94530-2602. A ASCII text file distributed by Internet, issue #1 was July 1993. E-mail: kcrosby@crayola.win.net.

Z-100 LifeLine, Steven W. Vagts, 2409 Riddick Rd. Elizabeth City, NC 27909, (919)338-8302. Publication for Z-100 (a S-100 machine).

The Staunch 8/89'er, Kirk L. Thompson editor, PO Box 548, West Branch IA 52358, (319)643-7136. $15/yr(US) publication for H-8/89s.

The SEBHC Journal, Leonard Geisler, 895 Starwick Dr., Ann Arbor MI 48105, (313)662-0750. Magazine of the Society of Eight-Bit Heath computerists, H-8 and H-89 support.

Sanyo PC Hackers Newsletter, Victor R. Frank editor, 12450 Skyline Blvd. Woodside, CA 94062-4541, (415)851-7031. Support for orphaned Sanyo computers and software.

the world of 68' micros, by FARNA Systems, PO Box 321, Warner Robins, GA 31099-0321. E-mail: dsrtfox@delphi.com. New magazine for support of old CoCo's and other 68xx(x) systems.

Amstrad PCW SIG, newsletter by Al Warsh, 2751 Reche Cyn Rd. #93, Colton, CA 92324. $9 for 6 bi-monthly newsletters on Amstrad CP/M machines.

Historically Brewed, A publication of the Historical Computer Society. Bimonthly at $18 a year. HCS, 2962 Park Street #1, Jacksonville, FL 32205. Editor David Greelish. Computer History and more.

IQLR (International QL Report), contact Bob Dyl, 15 Kilburn Ct. Newport, RI 02840. Subscription is $20 per year.

QL Hacker's Journal (QHJ), Timothy Swenson, 5615 Botkins Rd., Huber Heights, OH 45424, (513) 233-2178, sent mail & E-mail, swensotc@ss2.sews.wpafb.af.mil. Free to programmers of QL's.

Update Magazine, PO Box 1095, Peru, IN 46970, Subs $18 per year, supports Sinclair, Timex, and Cambridge computers.

Other Support Businesses

Hal Bower writes, sells, and supports B/PBios for Ampro, SB180, and YASBEC. $69.95. Hal Bower, 7914 Redglobe Ct., Severn MD 21144-1048, (410)551-5922.

Sydex, PO Box 5700, Eugene OR 97405, (503)683-6033. Sells several CP/M programs for use with PC Clones ('22Disk' format/copies CP/M disks using PC files system).

Elliam Associates, PO Box 2664, Atascadero CA 93423, (805)466-8440. Sells CP/M user group disks and Amstrad PCW products. See ad inside back cover.

Discus Distribution Services, Inc. sells CP/M for $150, CBASIC $600, Fortran-77 $350, Pascal/MT+ $600. 8020 San Miguel Canyon Rd., Salinas CA 93907, (408)663-6966.

Microcomputer Mail-Order Library of books, manuals, and periodicals in general and H/Zenith in particular. Borrow items for small fees. Contact Lee Hart, 4209 France Ave. North, Robbinsdale MN 55422, (612)533-3226.

Star-K Software Systems Corp. PO Box 209, Mt. Kisco, NY 10549, (914)241-0287, BBS: (914)241-3307. 6809/68000 operating system and software. Some educational products, call for catalog.

Peripheral Technology, 1250 E. Piedmont Rd., Marietta, GA 30067, (404)973-2156. 6809/68000 single board system. 68K ISA bus compatible system. See inside front cover.

Hazelwood Computers, RR#1, Box 36, Hwy 94@Bluffton, Rhineland, MO 65069, (314)236-4372. Some SS-50 6809 boards and new 68000 systems.

AAA Chicago Computers, Jerry Koppel, (708)681-3782. SS-50 6809 boards and systems. Very limited quanity, call for information.

MicroSolutions Computer Products, 132 W. Lincoln Hwy, DeKalb, IL 60115, (815)756-3411. Make disk copying program for CP/M systems, that runs on CP/M sytems, UNIFROM Format-translation. Also PC/Z80 CompatiCard and UniDos products.

GIMIX/OS-9, GMX, 3223 Arnold Lane, Northbrook, IL 60062, (800)559-0909, (708)559-0909, FAX (708)559-0942. Repair and support of new and old 6800/6809/68K/SS-50 systems.

n/SYSTEMS, Terry Hazen, 21460 Bear Creek Rd, Los Gatos CA 95030-9429, (408)354-7188, sells and supports the MDISK add-on RAM disk for the Ampro LB. PCB $29, assembled PCB $129, includes driver software, manual.

Corvatek, 561 N.W. Van Buren St. Corvallis OR 97330, (503)752-4833. PC style to serial keyboard adapter for Xerox, Kaypros, Franklin, Apples, $129. Other models supported.

Morgan, Thielmann & Associates services NON-PC compatible computers including CP/M as well as clones. Call Jerry Davis for more information (408) 972-1965.

Jim S. Thale Jr., 1150 Somerset Ave., Deerfield IL 60015-2944, (708)948-5731. Sells I/O board for YASBEC. Adds HD drives, 2 serial, 2 parallel ports. Partial kit $150, complete kit $210.

Trio Company of Cheektowaga, Ltd., PO Box 594, Cheektowaga NY 14225, (716)892-9630. Sells CP/M (& PC) packages: InfoStar 1.5 ($160); SuperSort 1.6 ($130), and WordStar 4.0 ($130).

Parts is Parts, Mike Zinkow, 137 Barkley Ave., Clifton NJ 07011-3244, (201)340-7333. Supports Zenith Z-100 with parts and service.

DYNACOMP, 178 Phillips Rd. Webster, NY 14580, (800)828-6772. Supplying versions of CP/M, TRS80, Apple, CoCo, Atari, PC/XT, software for older 8/16 bit systems. Call for older catalog.

The Computer Journal

Back Issues

Sales limited to supplies in stock.

Volume Number 1:
- Issues 1 to 9
- Serial interfacing and Modem transfers
- Floppy disk formats, Print spooler.
- Adding 8087 Math Chip, Fiber optics
- S-100 Hi-RES graphics.
- Controlling DC motors, Multi-user column.
- VIC-20 EPROM Programmer, CP/M 3.0.
- CP/M user functions and integration.

Volume Number 2:
- Issues 10 to 19
- Forth tutorial and Write Your Own.
- 68000 CPU for S-100.
- RPM vs CP/M, BIOS Enhancements.
- Poor Man's Distributed Processing.
- Controlling Apple Stepper Motors.
- Facsimile Pictures on a Micro.
- Memory Mapped I/O on a ZX81.

Volume Number 3:
- Issues 20 to 26
- Designing an 8035 SBC
- Using Apple Graphics from CP/M
- Soldering & Other Strange Tales
- Build an S-100 Floppy Disk Controller: WD2797 Controller for CP/M 68K
- Extending Turbo Pascal: series
- Unsoldering: The Arcane Art
- Analog Data Acquisition & Control: Connecting Your Computer to the Real World
- Programming the 8035 SBC
- NEW-DOS: series
- Variability in the BDS C Standard Library
- The SCSI Interface: series
- Using Turbo Pascal ISAM Files
- The Ampro Little Board Column: series
- C Column: series
- The Z Column: series
- The SCSI Interface: Introduction to SCSI
- Editing the CP/M Operating System
- INDEXER: Turbo Pascal Program to Create an Index
- Selecting & Building a System
- Introduction to Assembly Code for CP/M
- Ampro 186 Column
- ZTime-1: A Real Time Clock for the Ampro Z-80 Little Board

Volume Number 4:
- Issues 28 to 31
- Bus Systems: Selecting a System Bus
- Using the SB180 Real Time Clock
- The SCSI Interface: Software for the SCSI Adapter
- Inside Ampro Computers
- NEW-DOS: The CCP Commands (continued)
- ZSIG Corner
- Affordable C Compilers
- Concurrent Multitasking: A Review of DoubleDOS
- 68000 TinyGiant: Hawthorne's Low Cost 16-bit SBC and Operating System
- The Art of Source Code Generation: Disassembling Z-80 Software
- Feedback Control System Analysis: Using Root Locus Analysis & Feedback Loop Compensation
- The C Column: A Graphics Primitive Package
- The Hitachi HD64180: New Life for 8-bit Systems
- ZSIG Corner: Command Line Generators and Aliases
- A Tutor Program in Forth: Writing a Forth Tutor in Forth
- Disk Parameters: Modifying the CP/M Disk Parameter Block for Foreign Disk Formats
- Starting Your Own BBS
- Build an A/D Converter for the Ampro Little Board
- HD64180: Setting the Wait States & RAM Refresh using PRT & DMA
- Using SCSI for Real Time Control
- Open Letter to STD Bus Manufacturers
- Patching Turbo Pascal
- Choosing a Language for Machine Control
- Better Software Filter Design
- MDISK: Adding a 1 Meg RAM Disk to Ampro Little Board, Part 1
- Using the Hitachi hd64180: Embedded Processor Design
- 68000: Why use a new OS and the 68000?
- Detecting the 8087 Math Chip
- Floppy Disk Track Structure
- Double Density Floppy Controller
- ZCPR3 IOP for the Ampro Little Board
- 3200 Hackers' Language
- MDISK: Adding a 1 Meg RAM Disk to Ampro Little Board, Part 2
- Non-Preemptive Multitasking
- Software Timers for the 68000
- Lilliput Z-Node
- Using SCSI for Generalized I/O
- Communicating with Floppy Disks: Disk Parameters & their variations
- XBIOS: A Replacement BIOS for the SB180
- K-OS ONE and the SAGE: Demystifying Operating Systems
- Remote: Designing a Remote System Program
- The ZCPR3 Corner: ARUNZ Documentation

Issue Number 32:
- 15 copies now available -

Issue Number 33:
- Data File Conversion: Writing a Filter to Convert Foreign File Formats
- Advanced CP/M: ZCPR3PLUS & How to Write Self Relocating Code
- DataBase: The First in a Series on Data Bases and Information Processing
- SCSI for the S-100 Bus: Another Example of SCSI's Versatility
- A Mouse on any Hardware: Implementing the Mouse on a Z80 System
- Systematic Elimination of MS-DOS Files: Part 2, Subdirectories & Extended DOS Services
- ZCPR3 Corner: ARUNZ Shells & Patching WordStar 4.0

Issue Number 34:
- Developing a File Encryption System.
- Database: A continuation of the data base primer series.
- A Simple Multitasking Executive: Designing an embedded controller multitasking executive.
- ZCPR3: Relocatable code, PRL files, ZCPR34, and Type 4 programs.
- New Microcontrollers Have Smarts: Chips with BASIC or Forth in ROM are easy to program.
- Advanced CP/M: Operating system extensions to BDOS and BIOS, RSXs for CP/M 2.2.
- Macintosh Data File Conversion in Turbo Pascal.

Issue Number 35:
- All This & Modula-2: A Pascal-like alternative with scope and parameter passing.
- A Short Course in Source Code Generation: Disassembling 8088 software to produce modifiable assem. source code.
- Real Computing: The NS32032.
- S-100: EPROM Burner project for S-100 hardware hackers.
- Advanced CP/M: An up-to-date DOS, plus details on file structure and formats.
- REL-Style Assembly Language for CP/M and Z-System. Part 1: Selecting your assembler, linker and debugger.

Issue Number 36:
- Information Engineering: Introduction.
- Modula-2: A list of reference books.
- Temperature Measurement & Control: Agricultural computer application.
- ZCPR3 Corner: Z-Nodes, Z-Plan, Amstrand computer, and ZFILE.
- Real Computing: NS32032 hardware for experimenter, CPUs in series, software options.
- SPRINT: A review.
- REL-Style Assembly Language for CP/M & ZSystems, part 2.
- Advanced CP/M: Environmental programming.

Issue Number 37:
- C Pointers, Arrays & Structures Made Easier: Part 1, Pointers.
- ZCPR3 Corner: Z-Nodes, patching for NZCOM, ZFILER.
- Information Engineering: Basic Concepts: fields, field definition, client worksheets.
- Shells: Using ZCPR3 named shell variables to store date variables.
- Resident Programs: A detailed look at TSRs & how they can lead to chaos.
- Advanced CP/M: Raw and cooked console I/O.
- ZSDOS: Anatomy of an Operating System: Part 1.

Issue Number 38:
- C Math: Handling Dollars and Cents With C.
- Advanced CP/M: Batch Processing and a New ZEX.
- C Pointers, Arrays & Structures Made Easier: Part 2, Arrays.
- The Z-System Corner: Shells and ZEX, new Z-Node Central, system security under Z-Systems.
- Information Engineering: The portable Information Age.
- Computer Aided Publishing: Introduction to publishing and Desk Top Publishing.
- Shells: ZEX and hard disk backups.
- Real Computing: The National Semiconductor NS320XX.
- ZSDOS: Anatomy of an Operating System, Part 2.

Issue Number 39:
- Programming for Performance: Assembly Language techniques.
- Computer Aided Publishing: The Hewlett Packard LaserJet.
- The Z-System Corner: System enhancements with NZCOM.
- Generating LaserJet Fonts: A review of Digi-Fonts.
- Advanced CP/M: Making old programs Z-System aware.
- C Pointers, Arrays & Structures Made Easier: Part 3: Structures.
- Shells: Using ARUNZ alias with ZCAL.
- Real Computing: The National Semiconductor NS320XX.

Issue Number 40:
- Programming the LaserJet: Using the escape codes.
- Beginning Forth Column: Introduction.
- Advanced Forth Column: Variant Records and Modules.
- LINKPRL: Generating the bit maps for PRL files from a REL file.
- WordTech's dBXL: Writing your own custom designed business program.
- Advanced CP/M: ZEX 5.0=The machine and the language.
- Programming for Performance: Assembly language techniques.
- Programming Input/Output With C: Keyboard and screen functions.
- The Z-System Corner: Remote access systems and BDS C.
- Real Computing: The NS320XX

Issue Number 41:
- Forth Column: ADTs, Object Oriented Concepts.
- Improving the Ampro LB: Overcoming the 8MB hard drive limit.
- How to add Data Structures in Forth
- Advanced CP/M: CP/M is hacker's haven, and Z-System Command Scheduler.
- The Z-System Corner: Extended Multiple Command Line, and aliases.
- Programming disk and printer functions with C.
- LINKPRL: Making RSXes easy.
- SCOPY: Copying a series of unrelated files.

Issue Number 42:
- Dynamic Memory Allocation: Allocating memory at runtime with examples in Forth.
- Using BYE with NZCOM.
- C and the MS-DOS Screen Character Attributes.
- Forth Column: Lists and object oriented Forth.
- The Z-System Corner: Genie, BDS Z and Z-System Fundamentals.
- 68705 Embedded Controller Application: An example of a single-chip microcontroller application.
- Advanced CP/M: PluPerfect Writer and using BDS C with REL files.

Issue Number 43:
- Standardize Your Floppy Disk Drives.
- A New History Shell for ZSystem.
- Heath's HDOS, Then and Now.
- The ZSystem Corner: Software update service, and customizing NZCOM.
- Graphics Programming With C: Graphics routines for the IBM PC, and the Turbo C graphics library.
- Lazy Evaluation: End the evaluation as soon as the result is known.
- S-100: There's still life in the old bus.
- Advanced CP/M: Passing parameters, and complex error recovery.

Issue Number 44:
- Animation with Turbo C Part 1: The Basic Tools.
- Multitasking in Forth: New Micros F68FC11 and Max Forth.
- Mysteries of PC Floppy Disks Revealed: FM, MFM, and the twisted cable.
- DosDisk: MS-DOS disk format emulator for CP/M.
- Advanced CP/M: ZMATE and using lookup and dispatch for passing parameters.
- Forth Column: Handling Strings.
- Z-System Corner: MEX and telecommunications.

Issue Number 45:
- Embedded Systems for the Tenderfoot: Getting started with the 8031.
- The Z-System Corner: Using scripts with MEX.
- The Z-System and Turbo Pascal: Patching TURBO.COM to access the Z-System.
- Embedded Applications: Designing a Z80 RS-232 communications gateway, part 1.
- Advanced CP/M: String searches and tuning Jetfind.
- Animation with Turbo C: Part 2, screen interactions.
- Real Computing: The NS32000.

Issue Number 46:
- Build a Long Distance Printer Driver.
- Using the 8031's built-in UART for serial communications.
- Foundational Modules in Modula 2.
- The Z-System Corner: Patching The Word Plus spell checker, and the ZMATE macro text editor.
- Animation with Turbo C: Text in the graphics mode.
- Z80 Communications Gateway: Prototyping, Counter/Timers, and using the Z80 CTC.

Issue Number 47:
- Controlling Stepper Motors with the 68HC11F
- Z-System Corner: ZMATE Macro Language
- Using 8031 Interrupts
- T-1: What it is & Why You Need to Know
- ZCPR3 & Modula, Too
- Tips on Using LCDs: Interfacing to the 68HC705
- Real Computing: Debugging, NS32 Multitasking & Distributed Systems
- Long Distance Printer Driver: correction
- ROBO-SOG 90

Issue Number 48:

The Computer Journal Back Issues

- Fast Math Using Logarithms
- Forth and Forth Assembler
- Module-2 and the TCAP
- Adding a Bernoulli Drive to a CP/M Computer (Building a SCSI Interface)
- Review of BDS "Z"
- PMATE/ZMATE Macros, Pt. 1
- Real Computing
- Z-System Corner: Patching MEX-Plus and TheWord, Using ZEX

Issue Number 49:

- Computer Network Power Protection
- Floppy Disk Alignment w/RTXEB, Pt. 1
- Motor Control with the F68HC11
- Controlling Home Heating & Lighting, Pt. 1
- Getting Started in Assembly Language
- LAN Basics
- PMATE/ZMATE Macros, Pt. 2
- Z-System Corner/ Z-Best Software

Issue Number 50:

- Offload a System CPU with the Z181
- Floppy Disk Alignment w/RTXEB, Pt. 2
- Motor Control with the F68HC11
- Module-2 and the Command Line
- Controlling Home Heating & Lighting, Pt. 2
- Getting Started in Assembly Language Pt 2
- Local Area Networks
- Using the ZCPR3 IOP
- PMATE/ZMATE Macros, Pt. 3
- Z-System Corner, PCED/ Z-Best Software
- Real Computing, 32FX16, Caches

Issue Number 51:

- Introducing the YASBEC
- Floppy Disk Alignment w/RTXEB, Pt 3
- High Speed Modems on Eight Bit Systems
- A Z8 Talker and Host
- Local Area Networks—Ethernet
- UNIX Connectivity on the Cheap
- PC Hard Disk Partition Table
- A Short Introduction to Forth
- Stepped Inference in Embedded Control
- Real Computing, the 32CG160, Swordfish,
- PMATE/ZMATE Macros
- Z-System Corner, The Trenton Festival
- Z-Best Software, the Z3HELP System

Issue Number 52:

- YASBEC, The Hardware
- An Arbitrary Waveform Generator, Pt. 1
- B.Y.O. Assembler...in Forth
- Getting Started in Assembly Language, Pt. 3
- The NZCOM IOP
- Servos and the F68HC11
- Z-System Corner, Programming for Compatibility
- Z-Best Software
- Real Computing, X10 Revisited
- PMATE/ZMATE Macros
- Controlling Home Heating & Lighting, Pt. 3
- The CPU280, A High Performance Single-Board Computer

Issue Number 53:

- The CPU280
- Local Area Networks
- Am Arbitrary Waveform Generator
- Zed Fest '91
- Getting Started in Assembly Language
- The NZCOM IOP
- Z-BEST Software

Issue Number 54:

- B.Y.O. Assembler
- Local Area Networks
- Advanced CP/M
- ZCPR on a 16-Bit Intel Platform
- Real Computing
- Interrupts and the Z80
- 8 MHZ on a Ampro
- Hardware Heavenn
- What Zilog never told you about the Super8
- An Arbitary Waveform Generator
- The Development of TDOS

Issue Number 55:

- Fuzzilogy 101
- The Cyclic Redundancy Check in Forth
- The Internetwork Protocol (IP)
- Hardware Heaven
- Real Computing
- Remapping Disk Drives through the Virtual BIOS
- The Bumbling Mathmatician
- YASMEM
- Z-BEST Software

Issue Number 56:

- TCJ - The Next Ten Years
- Input Expension for 8031
- Connecting IDE Drives to 8-Bit Systems
- Real Computing
- 8 Queens in Forth
- Kaypro-84 Direct File Transfers
- Analog Signal Generation

Issue Number 57:

- Home Automation with X10
- File Transfer Protocols
- MDISK at 8 MHZ.
- Real Computing
- Shell Sort in Forth
- Introduction to Forth
- DR. S-100
- Z AT Last!

Issue Number 58:

- Multitasking Forth
- Computing Timer Values
- Affordable Development Tools
- Mr. Kaypro
- DR. S-100

Issue Number 59:

- Moving Forth
- Center Fold IMSAI MPU-A
- Developing Forth Applications
- Mr. Kaypro Review
- DR. S-100

Issue Number 60:

- Moving Forth Part II
- Center Fold IMSAI CPA
- Four for Forth
- Real Computing
- Debugging Forth
- Support Groups for Classics
- Mr. Kaypro Review
- DR. S-100

Issue Number 61:

- Multiprocessing 6809 part I
- Center Fold XEROX 820
- Quality Control
- Real Computing
- Support Groups for Classics
- Operating Systems - CP/M
- Mr. Kaypro 5MHZ

Issue Number 62:

- SCSI EPROM Programmer
- Center Fold XEROX 820
- DR S-100
- Moving Forth part III
- Programming the 6526 CIA
- Reminiscing and Musings
- Modem Scripts

Issue Number 63:

- SCSI EPROM Programmer part II
- Center Fold XEROX 820
- DR S-100
- Multiprocessing Part II
- 6809 Operating Systems
- Reminiscing and Musings
- IDE Drives Part II

Issue Number 64:

- Small-C?
- Center Fold last XEROX 820
- DR S-100
- Real Computing
- Moving Forth Part IV
- Small Systems
- Mr. Kaypro
- IDE Drives Part III

Issue Number 65:

- Small System Support
- Center Fold ZX80/81
- DR S-100
- Real Computing
- European Beat
- PC/XT Corner
- Little Circuits
- Levels of Forth
- Sinclair ZX81

Issue Number 66:

- Small System Support
- Center Fold: Advent Decoder
- DR S-100
- Connecting IDE Drives
- PC/XT Corner
- Little Circuits
- Multiprocessing Part III
- Z-System Corner

Issue Number 67:

- Small System Support
- Center Fold: SS-50/SS-30
- DR S-100
- Real Computing
- Serial Kaypro Interrupts
- Little Circuits
- Moving Forth Part 5
- European Beat

Issue Number 68:

- Small System Support
- Center Fold: Perisc/Mits 4PIO
- Z-System Corner II
- Real Computing
- PC/XT Corner
- Little Circuits
- Multiprocessing Forth Part 4
- Mr. Kaypro

Issue Number 69:

- Small System Support
- Center Fold: S-100 IDE
- Z-System Corner II
- Real Computing
- PC/XT Corner
- DR. S-100
- Moving Forth Part 6
- Mr. Kaypro

Issue Number 70:

- Small System Support
- Center Fold: Jupiter ACE
- Z-System Corner II
- Real Computing
- PC/XT Corner: Stepper Motors
- DR. S-100
- Multiprocessing Part 5
- European Beat

Issue Number 71:

- Computing Hero of 1994
- Small System Support
- Center Fold: Hayes 80-103A
- Power Supply Basics
- Real Computing
- PC/XT Corner: Stepper Motors
- DR. S-100
- Moving Forth Part 7
- Mr. Kaypro
- 8048 Emulator Part 1

Issue Number 72

- Beginning PLD
- Small System Support
- Center Fold: Rockwell R65F11
- Playing With Micros
- Real Computing
- Small Tools Part 1
- DR. S-100
- Moving Forth Part 7.5
- 8048 Emulator Part 2

SPECIAL DISCOUNT

15% on cost of Back Issues when buying from 1 to Current Issue.
10% on 10 or more issues.

Subscriptions (CA not taxable)	U.S.	Canada/Mexico (Surface)	Canada/Mexico (Air)	Europe/Other (Surface)	Europe/Other (Air)
1year (6 issues)	$24.00	$32.00	$34.00	$34.00	$44.00
2 years (12 issues)	$44.00	$60.00	$64.00	$64.00	$84.00
Back issues (CA tax)	add these shipping costs for each issue ordered				
Bound Volumes $20.00 ea	+$3.00	+$3.50	+$6.50	+$4.00	+$17.00
#20 thru #43 are $3.00 ea.	+$1.00	+$1.00	+$1.25	+$1.50	+$2.50
#44 and up are $4.00ea.	+$1.25	+$1.25	+$1.75	+$2.00	+$3.50

Items: _____

_____

_____ Back Issues Total _____

_____ Shipping Total _____

California state Residents add 7.25% Sales TAX _____

Subscription Total _____

Total Enclosed _____

Name: _____

Address: _____

_____

_____

_____

Credit Card # _____-_____-_____-_____ exp ___/___

Payment is accepted by check, money order, or Credit Card (M/C, VISA, CarteBlanche, Diners Club). Checks must be in US funds, drawn on a US bank. Credit Card orders can call 1(800) 424-8825.

TCJ The Computer Journal

P.O. Box 535, Lincoln, CA 95648-0535
Phone (916) 645-1670

Regular Feature

Editorial Comment

Trouble shooting XTs

The Computer Corner

By Bill Kibler

Welcome to our special on the PC/XT machine. I covered most of the concepts you need in the special and centerfold. I do get plenty of calls and just yesterday, I helped a friend fix his machine over the phone. With that in mind, I thought it might help if I covered the how-to of trouble shooting a system.

POWER

When working on any system the first place you start after a visual inspection is checking the power system out. I say after the visual inspection, since sticking the power cable back in the wall socket is certainly easier than any of the other steps I will outline. Your visual inspection is mostly to find glaring problems (like not plugged in) and get a feel for the system. After that we want to make sure you have power and the power supply is working properly.

The way I checkout the power is to turn it on and listen. With the older MFM hard drives, you could hear them ramping up to speed, sort of like listening to a jet engine taking off. Often when the power supply has problems, the older hard drives will have trouble reaching their normal speed and may stop, slow back down, or start hunting (going faster then slower then faster). There can be other reasons for that speed problem, but most often it is a bad supply or one that is overloaded.

Another sound to listen for is the fan speed. Oh yes, we have had a couple of cases where the newer 486 CPU cooling fans have gone bad, sounding like power supply fan problems. Take the cover off and listen and look at where bad sounds are coming from. Often lots of dust will be binding the fan up and some cleaning with a dry brush or vacuum will return things to normal.

Now you can have a part of a supply go bad as well. I had several older supplies in which there was a separate regulator circuit for each pair of cables. To find the bad one required putting a load on the circuit (like a floppy unit) and checking the voltage. You should see 5 and 12 volts, but might see 8, not 12, if it is not regulating properly. I have rarely seen overvoltage, but it can happen and burn out all components on your computer.

Most modern power supplies however have what is called a crowbar circuit. This circuit is suppose to shut down the power supply if a short or other damaging problems happen. So you might take a new supply and stick it in as a replacement only to still have nothing happen. You proably have a short somewhere and the supply is just shutting down. You can check this out by removing loads (undoing conectors) and then retrying the power. If your fan starts again, put connectors back on one at a time till you find the shorted cable. I don't beleive any PC power supplies will shut down without a load, as the fan current should keep the crowbar circuit from kicking in (some crowbars shutdown if load goes to zip.)

What Next

After checking and being sure your power supply is working, I like to check for system activity. Often I simply see if the fans and drives run at proper speed and then see if it tries to boot. Most BIOS programs will go through a number of steps that are visible and make sounds. Stepping the drives home (their normal starting position) is one sign the BIOS was able to run and thus the power and motherboard are proably working ok. I say proably, only because you might have a bad power cable to the hard drive and yet the floppy works fines.

It is at this point that we can start running into configuration problems. As I stated in my $10 XT article, a common problem can be a misconfigured system. The PC/XT systems use dip switches for setting what type of options you may have. On AT and later units, a menu driven utility stores system parameters. The newer options can cause many problems if set wrong. One option I sometimes do is to not have the A: drive be checked for boot disk. Thus watching for the BIOS to step and try to load from A drive might not be valid on one so set up. On 386 and 486 systems, there are many options having to do with how the CPU chip handles memory, keyboards, video ram, and 20 or 30 other possible settings, anyone of which might be a reason for not working properly. Set these options with care, write them down in case they get trashed, especially your hard drive settings.

I think all setup programs have an default option which should give you a safe starting point if yours becomes too badly messed up. For configuration problems such as wrong drive type, you will get an error message. Most will allow you to boot up still, but you must hit the F1 key to continue on. This will also give a false problem indication, because you might have an error that stops things before it steps drives. In my earlier case, the switch setting were wrong for monitor type and thus I did not have anyway to see the message. I blindly hit the F1 key anyway, and then my drives got stepped. That told me the system was running but

with some error and possibly a misconfigured video option.

The solution is first to get video and you do that by changing the switch setting and repowering up the system. Suppose the switch is correct, then a bad video board or monitor is possible. To check this, is to replace the video board with a new or know working unit. I usually have two or three other systems setting around, so using spares is always possible for me.

RECORD!

While on the topic, it is very important to record, ON PAPER, what your hard disk setup values are. Most newer IDE drives have the setup values on the drive, but the older MFM and even some IDE drives did not. Even the newer IDEs have some latitude for setting their parameters and should you use non-standard values, these need to be recorded. There are many free text files that contain a fairly long and accurate list of what values of size, heads, and cyclinders most of the manufactured drives are. However it can be a real big problem getting data back off a crashed drive even if you know the values.

Some of the early MFM drive controllers have their own hard disk initialization routines and thus the drive setup information is contained somewhere on the disk. The different programs can have this information stored in different places and in different ways. So if you have an MFM type drive, chances are a drive controller problem might mean you are unable to retrieve the data from the working hard drive, without an identical controller.

I remember having to reformat hard drives each time I tested out a controller card. Each one had their own protocol, and yet they all seemed to format the same. You do a low level format from DEBUG, by going G=C800:5 <return>. That jumps you into the configuration and format portion of the on board (controller board) BIOS ROM. Now if you want to upgrade you PC/XT to a newer IDE drive, which most likely will be a big step up, you will need to get an IDE controller that has it's own BIOS and of course is also a 8 bit card. The 16 bit cards are all for AT class machines and will not work in an older PC/XT.

I have one of the BIOS optional cards and they can sense the hard drive parameters and allow you to boot them. I recently went to a 1.2Gig drive on my main system, only to discover that the BIOS can't really handle that large of a drive. Most brand new BIOS's will have updated code for the larger drives, but most older BIOS settings have limits of 16 heads, or 1024 cylinders, and 64 sectors as their maximums.

A Good buy!

A very good program to buy that comes with the bigger hard drives, is Ontracks's Disk Manager. I first used this program when I was installing a Novel network on a non-standard hard drive in a 286 system. The Disk Manager program can firgure out what your drive type is and then set up drivers and other needed programs so you can use the drive. My Gig drive's boot sectors now contain a special program that makes the BIOS able to read the hard drive. For repairing many of the older MFM drives, you might find a version of Disk Manger your ideal tool.

A last note on drives is watch out for block size. You might run into problems when moving to the larger drives. Both DOS and CP/M have only so many bits set aside for block allocation. That means as the drive size gets larger the block size must also get larger, or you will not be able to utilize the entire disk. Generally under DOS it goes like this, below a 128 megs it is 2K blocks, 129 to 256 is 4K blocks, 257 to 512 megs are 8K blocks, and 513 to 1 gig is 16K blocks. Above one gig you have to use 32K blocks, which means the smallest file will take up 32K of space. So even a 2 byte record uses 32K if you leave your hard drive in one partition.

I always recommend breaking drives into multiple partitions or false drives. You should keep you boot drive to under 256 megs if you can, thus all the very small startup files will only use 4K blocks.

Lastly

There is a whole passel of things that can still go wrong, you can have bad connectors on interfaces, wrong jumpers for the memory you have just added, overheating CPU (although it usually isn't the problem), a bad com port (but the other one is good), and all of these will drive you crazy finding them. The best approach is to obtain some good software and hardware tools. Practice a little snooping around while your system is running OK and see how to use the tools. Most newer MSDOS machines have MSD, which is Microsoft System Diagnostics. This tool will give you access to most parameters and minor tests of ports. There are better programs that will fully test memory, com ports, video screens, and about any option you might have.

People willing to get in their and hunt around, usually develop a series of personal tools and skills that speed up the trouble shooting process. I once came into a work location and found a note from the person who was on duty the night before. He had spent the entire night trying to figure out what had failed on one machine. He gave up and went home. I had plenty of repair time on the machine and he was sure I would fix it. It still blew him away when I told him how long it took me to find the problem, about 30 seconds. The machine has a reset switch and my first testing procedure is to reset the system. When I pushed the reset switch it didn't feel just right. A quick check with an Ohm meter showed the switch in fact was shorted and thus the system was in constant reset mode.

The basic skill I used was a strong personal knowledge of how the system worked and how it *feels* when you use the devices. You too need to develop that sense of understanding, and then trouble shooting will no longer be a black art, but a skill you have mastered.

That is all the advice I have time for, so send me those questions and letters, and keep hacking! Bill.

TCJ CLASSIFIED

CLASSIFIED RATES!
$5.00 PER LISTING!

TCJ Classified ads are on a prepaid basis only. The cost is $5.00 per ad entry. Support wanted is a free service to subscribers who need to find old or missing documentation or software. Please limit your requests to one type of system.

Commercial Advertising Rates:

Size	Once	4+
Full	$120	$90
1/2 Page	$75	$60
1/3 Page	$60	$45
1/4 Page	$50	$40
Market Place	$25	$100/yr

Send your items to:
The Computer Journal
P.O. Box 535
Lincoln, CA 95648-0535

Historically Brewed. The magazine of the Historical Computer Society. Read about the people and machines which changed our world. Buy, sell and trade "antique" computers. Subscriptions $18, or try an issue for $3. HCS, 2962 Park Street #1, Jacksonville, FL 32205

Start your own technical venture! Don Lancaster's newly updated **INCREDIBLE SECRET MONEY MACHINE II** tells how. We now have autographed copies of the Guru's underground classic for $21.50. Synergetics Press, Box 809-J, Thatcher AZ, 85552.

THE CASE AGAINST PATENTS Throughly tested and proven alternatives that work in the real world. $33.50. Synergetics Press, Box 809-J, Thatcher AZ, 85552.

SELL: IMSAI 8080 system, serial #4605, completely assembled and fully operational. Has many boards: IMSAI, Seals, Vectorgrphic, Tarbell, Polymorphic, Mullen and Dubois. Includes Microplois 1053 Dual Floppy Drives with controller board. Sanyo B&W monitor. Teletype 33ASR printer. Every item has COMPLETE manual and diagrams. Cassette bootstrap. Card extender test board. Software. Many parts and extras. Constructed 1976. Len Silvern, Box 2085, Sedona AZ 86339; (520) 282-3009.

Sell: Floppy disks, 10 unused 5 1/4" hard sectored, 16 sector disks. $5.00 will cover shipping and handling. Call Bob at 415-967-7559 (W6NBI).

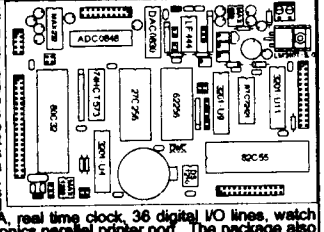
TCJ ADS WORK!

Classified ads in *TCJ*
get results, FAST!
Need to sell that special older
system - TRY *TCJ*.
World Wide Coverage
with Readers interested in what
YOU have to sell.
Provide a support service,
our readers are looking for
assistance with their older
systems - all the time.
The best deal in magzines,
TCJ Classified
it works!

Wanted. Books/Manuals/Info on a Structured Design SD20/24 PAL development system. Has built in monitor, but unable to read pre-programmed PALs. Anyone know if these people still in business? Contact Bill at *TCJ* (800)424-8825.

Wanted: Televideo terminal. Model 905, 924, 955, 9550 or 970. Alen Gordon MD, 160 N.W. 176 St.. Miami, FL 33169 (305) 653-8000.

VERSATILE 80C32 BASED SINGLE BOARD COMPUTER



The DC8032-1 is the first in a series of versatile single board computers based on the 8051/52 family of microcontrollers. The DC8032-1 and DC8032-2, available later this year, has been designed to provide most, if not all, of the functions required of a dedicated controller. Standard features include 32K of RAM & EPROM, A/D & D/A, real time clock, 36 digital I/O lines, watch dog timer and a centronics parallel printer port. The package also includes extended BASIC-52 with 16 additional commands, a debug monitor with 12 commands and DC_TERM, a terminal emulator for communicating with the DC8032-1. A 9-volt DC wall cube is also included. SPECIAL PRICING FOR STUDENTS

$200.00 ASSEMBLED AND ESTED, $175.00 KIT
$5.00 Shipping & Handeling + $5.00 for C.O.D.
SEND CASHIERS CHECK, MO. OR C.O.D. TO:
D. C. MICROS 1843 SUMNER CT.
LAS CRUCES, NM 88001 PH. (505) 524-4029

DIBs Electronic Design

Dave Baldwin

6619 Westbrook Dr. Voice/Fax (916) 722-3877
Citrus Heights, CA 95621 DIBs BBS (916) 722-5799

Discover
The Z-Letter
The Z-letter is the only publication exclusively for CP/M and the Z-System. Eagle computers and Spellbinder support. Licensed CP/M distributor.

Subscriptions: $18 US, $22 Canada and Mexico, $36 Overseas. Write or call for free sample.

The Z-Letter
Lambda Software Publishing
149 West Hilliard Lane
Eugene, OR 97404-3057
(503) 688-3563

Advent Kaypro Upgrades
TurboROM. Allows flexible configuration of your entire system, read/write additional formats and more, only $35.

Replacement Floppy drives and Hard Drive Conversion Kits. Call or write for availability & pricing.

Call (916)483-0312
eves, weekends or write
Chuck Stafford
4000 Norris Ave.
Sacramento, CA 95821

TCJ MARKET PLACE
Advertising for small business
First Insertion: $25
Reinsertion: $20
Full Six issues $100
Rates include typesetting.
Payment must accompany order.
VISA, MasterCard, Diner's Club, Carte Blanche accepted.
Checks, money orders must be US funds. Resetting of ad consitutes a new advertisement at first time insertion rates.
Mail ad or contact
The Computer Journal
P.O. Box 535
Lincoln, CA 95648-0535

CP/M SOFTWARE

100 page Public Domain Catalog, $8.50 plus $1.50 shipping and handling. New Digital Research CP/M 2.2 manual, $19.95 plus $3.00 shipping and handling. Also, MS/PC-DOS Software. Disk Copying, including AMSTRAD. Send self addressed, stamped envelope for free Flyer, Catalog $1.00

Elliam Associates
Box 2664
Atascadero, CA 93423
805-466-8440

MORE POWER!
68HC11, 80C51 & 80C166
☑ More Microcontrollers.
☑ Faster Hardware.
☑ Faster Software.
☑ More Productive.
☑ More Tools and Utilities.
Low cost SBC's from $84. Get it done today! Not next month.
For brochure or applications:
AM Research
4600 Hidden Oaks Lane
Loomis, CA 95650
1(800)947-8051
sofia@garlic.com

S-100/IEEE-696
IMSAI Altair
Compupro Morrow
Cromemco
and more!
Cards • Docs • Systems
Dr. S-100
Herb Johnson,
CN 5256 #105,
Princeton, NJ 08543
(609) 771-1503
hjohnson@pluto.njcc.com

THE FORTH SOURCE
Hardware & Software
MOUNTAIN VIEW PRESS
Glen B. Haydon, M.D.
Route 2 Box 429
La Honda, CA 94020
(415) 747 0760

PCB's in Minutes From LaserPrint!*
8 1/2" x 11" Sheets 100% MBG
* Or Photocopier Use household iron to apply.
Press-n-Peel
PnP BLUE or **PnP WET**
For High Precision Professional PCB Layouts
1. LaserPrint
2. Iron-On
3. Peel-Off
4. Etch
An Extra Layer of Resist for Super Fine Traces
Easy Hobby Quality PCB's
1. LaserPrint
2. Iron-On
3. Soak-Off w/ Water
4. Etch
Transfers Laser or Copier Toner as Resist
20Sh$30/40Sh$50/100Sh$100 Blue/Wet (No Mix)
Sample Pack 5 Shts Blue + 5 Shts Wet $20
VISA/MC/PO/CK/MO $4 S&H -- 2nd Day Mail
Techniks Inc. P.O. Box 463 Ringoes NJ 08551
(908)788-8249